



IP Addresses and Services Configuration Guide for Cisco CRS Routers, IOS XR Release 6.7.x

First Published: 2020-08-01

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 NON-INFRINGEMENT 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.

The documentation set for this product strives to use bias-free language. For purposes of this documentation set, bias-free is defined as language that does not imply discrimination based on age, disability, gender, racial identity, ethnic identity, sexual orientation, socioeconomic status, and intersectionality. Exceptions may be present in the documentation due to language that is hardcoded in the user interfaces of the product software, language used based on standards documentation, or language that is used by a referenced third-party product.

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)

© 2020 Cisco Systems, Inc. All rights reserved.



CONTENTS

PREFACE

Preface xv

Changes to This Document xv

Communications, Services, and Additional Information xv

CHAPTER 1

New and Changed IP Addresses and Services Features 1

IP Addresses and Services Features Added or Modified in IOS XR Release 6.7.x 1

CHAPTER 2

Implementing Network Stack IPv4 and IPv6 3

Prerequisites for Implementing Network Stack IPv4 and IPv6 4

Restrictions for Implementing Network Stack IPv4 and IPv6 4

Information About Implementing Network Stack IPv4 and IPv6 4

Network Stack IPv4 and IPv6 Exceptions 4

IPv4 and IPv6 Functionality 4

IPv6 for Cisco IOS XR Software 5

Larger IPv6 Address Space 5

IPv6 Address Formats 5

IPv6 Address Type: Unicast 6

Aggregatable Global Address 7

Link-Local Address 8

IPv4-Compatible IPv6 Address 8

IPv6 Address Type: Multicast 9

Simplified IPv6 Packet Header 10

IPv6 Neighbor Discovery 13

IPv6 Neighbor Solicitation Message 13

IPv6 Router Advertisement Message 15

IPv6 Neighbor Redirect Message 16

| | |
|---|----|
| ICMP for IPv6 | 18 |
| Address Repository Manager | 18 |
| Address Conflict Resolution | 18 |
| Route-Tag Support for Connected Routes | 19 |
| How to Implement Network Stack IPv4 and IPv6 | 20 |
| Assigning IPv4 Addresses to Network Interfaces | 20 |
| IPv4 Addresses | 20 |
| Configuring IPv6 Addressing | 22 |
| IPv6 Multicast Groups | 22 |
| IPv6 Virtual Addresses | 24 |
| Assigning Multiple IP Addresses to Network Interfaces | 24 |
| Secondary IPv4 Addresses | 24 |
| Configuring IPv4 and IPv6 Protocol Stacks | 25 |
| Enabling IPv4 Processing on an Unnumbered Interface | 26 |
| IPv4 Processing on an Unnumbered Interface | 26 |
| Configuring ICMP Rate Limiting | 28 |
| IPv4 ICMP Rate Limiting | 28 |
| IPv6 ICMP Rate Limiting | 28 |
| Selecting Flexible Source IP | 29 |
| Configuring IPARM Conflict Resolution | 30 |
| Static Policy Resolution | 30 |
| Longest Prefix Address Conflict Resolution | 31 |
| Highest IP Address Conflict Resolution | 31 |
| Generic Routing Encapsulation | 32 |
| IPv4 Forwarding over GRE Tunnels | 32 |
| Configuration Examples for Implementing Network Stack IPv4 and IPv6 | 33 |
| Assigning an Unnumbered Interface: Example | 33 |
| Additional References | 33 |

CHAPTER 3
Implementing ARP 35

| | |
|-----------------------------------|----|
| Prerequisites for Configuring ARP | 35 |
| Restrictions for Configuring ARP | 35 |
| Information About Configuring ARP | 36 |
| IP Addressing Overview | 36 |

| | |
|--|--|
| Address Resolution on a Single LAN | 36 |
| Address Resolution When Interconnected by a Router | 36 |
| ARP and Proxy ARP | 37 |
| ARP Cache Entries | 38 |
| How to Configure ARP | 38 |
| Defining a Static ARP Cache Entry | 38 |
| Enabling Proxy ARP | 39 |
| Enabling Local Proxy ARP | 39 |
| Configure Learning of Local ARP Entries | 40 |
| Additional References | 41 |
| <hr/> | |
| CHAPTER 4 | Implementing the Dynamic Host Configuration Protocol 43 |
| Prerequisites for Configuring DHCP Relay Agent | 43 |
| Information About DHCP Relay Agent | 44 |
| Secure ARP | 44 |
| How to Configure and Enable DHCP Relay Agent | 44 |
| Configuring and Enabling DHCP Relay Agent with DHCP MAC Address Verification | 45 |
| Configuring the DHCPv6 (Stateless) Relay Agent | 46 |
| Enabling DHCP Relay Agent on an Interface | 47 |
| Disabling DHCP Relay on an Interface | 47 |
| Enabling DHCP Relay on a VRF | 48 |
| Configuring the Relay Agent Information Feature | 48 |
| Configuring Relay Agent Giaddr Policy | 50 |
| DHCPv4 Client | 51 |
| Enabling DHCP Client on an Interface | 52 |
| Information About Configuring DHCP IPv6 Information Pools | 52 |
| How to Configure DHCP IPv6 Information Pools | 52 |
| Configuring Cisco IOS XR DHCP IPv6 Information Pool Option | 52 |
| Enabling Secure ARP | 53 |
| Configuration Examples for the DHCP Relay Agent | 54 |
| DHCP Relay Profile: Example | 54 |
| DHCP Relay on an Interface: Example | 54 |
| DHCP Relay on a VRF: Example | 54 |
| Relay Agent Information Option Support: Example | 55 |

| | |
|---|----|
| Relay Agent Giaddr Policy: Example | 55 |
| Cisco IOS XR Broadcast Flag Policy: Example | 55 |
| Additional References | 56 |

CHAPTER 5**Implementing Host Services and Applications 59**

| | |
|---|----|
| Prerequisites for Implementing Host Services and Applications | 59 |
| Information About Implementing Host Services and Applications | 60 |
| Key Features Supported in the Cisco IOS XR software Host Services and Applications Implementation | 60 |
| Network Connectivity Tools | 60 |
| Ping | 60 |
| Traceroute | 61 |
| Domain Services | 61 |
| TFTP Server | 61 |
| File Transfer Services | 62 |
| RCP | 62 |
| FTP | 62 |
| TFTP | 62 |
| SCP | 63 |
| Cisco inetd | 63 |
| Telnet | 63 |
| How to Implement Host Services and Applications | 63 |
| Checking Network Connectivity | 63 |
| Checking Network Connectivity for Multiple Destinations | 64 |
| Checking Packet Routes | 65 |
| Configuring Domain Services | 65 |
| Configuring a Router as a TFTP Server | 66 |
| Configuring a Router to Use rcp Connections | 67 |
| Configuring a Router to Use FTP Connections | 69 |
| Configuring a Router to Use TFTP Connections | 71 |
| Configuring Telnet Services | 72 |
| Transferring Files Using SCP | 72 |
| Configuring syslog source-interface | 73 |
| IPv6 Support for IP SLA ICMP Echo Operation | 74 |

| | |
|--|----|
| Configuring an IPSLA ICMP echo operation | 74 |
| Configuration Examples for Implementing Host Services and Applications | 75 |
| Checking Network Connectivity: Example | 75 |
| Configuring Domain Services: Example | 76 |
| Configuring a Router to Use rep, FTP, or TFTP Connections: Example | 77 |
| Additional References | 78 |

CHAPTER 6

| | |
|---|-----------|
| Prerequisites for Implementing Access Lists and Prefix Lists | 79 |
| Restrictions for Implementing Access Lists and Prefix Lists | 79 |
| Restrictions for Implementing ACL-based Forwarding | 80 |
| Information About Implementing Access Lists and Prefix Lists | 80 |
| Access Lists and Prefix Lists Feature Highlights | 80 |
| Purpose of IP Access Lists | 81 |
| How an IP Access List Works | 81 |
| IP Access List Process and Rules | 81 |
| Helpful Hints for Creating IP Access Lists | 82 |
| Source and Destination Addresses | 83 |
| Wildcard Mask and Implicit Wildcard Mask | 83 |
| Transport Layer Information | 83 |
| IP Access List Entry Sequence Numbering | 83 |
| Sequence Numbering Behavior | 83 |
| Understanding IP Access List Logging Messages | 84 |
| Extended Access Lists with Fragment Control | 85 |
| Policy Routing | 87 |
| Comments About Entries in Access Lists | 87 |
| Access Control List Counters | 87 |
| BGP Filtering Using Prefix Lists | 88 |
| How the System Filters Traffic by Prefix List | 88 |
| Information About Implementing ACL-based Forwarding | 89 |
| ACL-based Forwarding Overview | 89 |
| ACL-based Forwarding Functions | 89 |
| How to Implement Access Lists and Prefix Lists | 89 |
| Configuring Extended Access Lists | 90 |
| Applying Access Lists | 92 |

- Controlling Access to an Interface 92
- Controlling Access to a Line 93
- Configuring Prefix Lists 94
- Configuring Standard Access Lists 96
- Copying Access Lists 97
- Sequencing Access-List Entries and Revising the Access List 98
- Copying Prefix Lists 100
- Sequencing Prefix List Entries and Revising the Prefix List 101
- How to Implement ACL-based Forwarding 102
 - Configuring ACL-based Forwarding with Security ACL 102
 - Configuring Pure ACL-Based Forwarding for IPv6 ACL 104
 - Configuring Pure ACL-based Forwarding for ACL 105
 - ACL-Chaining 106
 - ACL-Chaining Overview 106
 - Restrictions for Common ACL 106
 - Configuring an Interface to accept Common ACL 107
- Configuration Examples for Implementing Access Lists and Prefix Lists 108
 - Resequencing Entries in an Access List: Example 108
 - Adding Entries with Sequence Numbers: Example 108
 - Adding Entries Without Sequence Numbers: Example 109
- Configuration Examples for Implementing ACL-based Forwarding 109
 - ACL with Security and ACL-based Forwarding Access Control Entry: Example 109
 - Pure ACL-based Forwarding for ACL Example 110
 - IPv6 ACL-based Forwarding Example 110
- IPv6 ACL in Class Map 111
 - Configuring IPv6 ACL QoS - An Example 112
- Configuring an Interface to accept Common ACL - Examples 114
- Additional References 116

CHAPTER 7

Implementing Video Monitoring 119

- Prerequisites for Implementing Video Monitoring 119
- Information About Implementing Video Monitoring 119
 - Video Monitoring 119
 - Introduction to Video Monitoring 120

| | |
|--|-----|
| Key Features Supported on Video Monitoring | 120 |
| Video Monitoring Terminology | 123 |
| Implementing Video Monitoring | 124 |
| Creating IPv4 Access Lists | 124 |
| Configuring class-map | 125 |
| Configuring policy-map | 126 |
| Configuring policy-map with metric parameters | 126 |
| Configuring policy-map with flow parameters | 129 |
| Configuring policy-map with react parameters | 130 |
| Configuring service policy on an interface | 132 |
| Configuration Examples for Implementing Video Monitoring | 133 |
| Additional References | 138 |

CHAPTER 8
Implementing Cisco Express Forwarding 139

| | |
|--|-----|
| Prerequisites for Implementing Cisco Express Forwarding | 139 |
| Information About Implementing Cisco Express Forwarding Software | 140 |
| Key Features Supported in the Cisco Express Forwarding Implementation | 140 |
| Benefits of CEF | 140 |
| CEF Components | 140 |
| Border Gateway Protocol Policy Accounting | 141 |
| Reverse Path Forwarding (Strict and Loose) | 142 |
| Route Processor Management Ethernet Forwarding | 143 |
| Per-Flow Load Balancing | 143 |
| BGP Attributes Download | 144 |
| GTP Tunnel Load Balancing | 145 |
| How to Implement CEF | 145 |
| Verifying CEF | 145 |
| Configuring BGP Policy Accounting | 146 |
| Verifying BGP Policy Accounting | 147 |
| Configuring a Route Purge Delay | 149 |
| Configuring Unicast RPF Checking | 149 |
| Configuring Modular Services Card-to-Route Processor Management Ethernet Interface Switching | 150 |
| Configuring Per-Flow Load Balancing | 151 |
| Configuring a 7-Tuple Hash Algorithm | 151 |

| | |
|--|-----|
| Verifying the CEF Exact Route with 7-Tuple Parameters | 152 |
| Configuring BGP Attributes Download | 153 |
| Configuring BGP Attributes Download | 153 |
| Configuration Examples for Implementing CEF on Routers Software | 153 |
| Configuring BGP Policy Accounting: Example | 154 |
| Verifying BGP Policy Statistics: Example | 157 |
| Configuring Unicast RPF Checking: Example | 170 |
| Configuring the Switching of Modular Services Card to Management Ethernet Interfaces on the Route Processor: Example | 170 |
| Configuring Per-Flow Load Balancing: Example | 170 |
| Configuring BGP Attributes Download: Example | 171 |
| Configuring GTP Tunnel Load Balancing: Example | 171 |
| Additional References | 173 |

CHAPTER 9**Implementing EPFT 175**

| | |
|--|-----|
| Information About EPFT | 175 |
| Prerequisites for Implementing EPFT | 176 |
| Restrictions for Implementing EPFT | 176 |
| Enabling Excessive Punt Flow Trap Processing | 176 |
| Additional References | 178 |

CHAPTER 10**Implementing HSRP 181**

| | |
|-------------------------------------|-----|
| Prerequisites for Implementing HSRP | 182 |
| Restrictions for Implementing HSRP | 182 |
| Information About Implementing HSRP | 182 |
| HSRP Overview | 182 |
| HSRP Groups | 182 |
| HSRP and ARP | 184 |
| Preemption | 185 |
| ICMP Redirect Messages | 185 |
| How to Implement HSRP | 185 |
| Enabling HSRP | 185 |
| Enabling HSRP for IPv6 | 187 |
| Configuring HSRP Group Attributes | 188 |

| | |
|---|-----|
| Configuring the HSRP Activation Delay | 192 |
| Enabling HSRP Support for ICMP Redirect Messages | 193 |
| Multiple Group Optimization (MGO) for HSRP | 195 |
| Customizing HSRP | 195 |
| Configuring a Primary Virtual IPv4 Address | 197 |
| Configuring a Secondary Virtual IPv4 Address | 199 |
| Configuring the Subordinate Group to Inherit its State from a Specified Group | 200 |
| Configuring a Subordinate Primary Virtual IPv4 Address | 201 |
| Configuring a Secondary Virtual IPv4 address for the Subordinate Group | 202 |
| Configuring a Subordinate Virtual MAC Address | 203 |
| Configuring an HSRP Session Name | 204 |
| BFD for HSRP | 205 |
| Advantages of BFD | 205 |
| BFD Process | 205 |
| Configuring BFD | 205 |
| Enabling BFD | 206 |
| Modifying BFD timers (minimum interval) | 206 |
| Modifying BFD timers (multiplier) | 207 |
| Enhanced Object Tracking for HSRP and IP Static | 208 |
| Configuring object tracking for HSRP | 209 |
| Hot Restartability for HSRP | 210 |
| Configuration Examples for HSRP Implementation on Software | 210 |
| Configuring an HSRP Group: Example | 210 |
| Configuring a Router for Multiple HSRP Groups: Example | 210 |
| Additional References | 211 |

CHAPTER 11
Implementing LPTS 213

| | |
|---|-----|
| Prerequisites for Implementing LPTS | 213 |
| Information About Implementing LPTS | 213 |
| LPTS Overview | 213 |
| LPTS Policers | 214 |
| Configuring LPTS Policers | 214 |
| Configuration Examples for Implementing LPTS Policers | 215 |
| Configuring LPTS Policers: Example | 216 |

Additional References 220

CHAPTER 12**Implementing VRRP 223**

- Prerequisites for Implementing VRRP on Cisco IOS XR Software 224
- Restrictions for Implementing VRRP on Cisco IOS XR Software 224
- Information About Implementing VRRP 224
 - VRRP Overview 224
 - Multiple Virtual Router Support 225
 - VRRP Router Priority 226
 - VRRP Advertisements 226
 - Benefits of VRRP 226
- Configuring VRRP 227
 - Configuring VRRP for IPv4 Networks 227
 - Configuring VRRP for IPv6 Networks 229
 - Clearing VRRP Statistics 231
 - Disabling State Change Logging 232
- Multiple Group Optimization for Virtual Router Redundancy Protocol 232
 - Configuring a VRRP Session Name 233
 - Configuring the Subordinate Group to Inherit its State from a Specified Group (VRRP) 234
 - Configuring a Primary Virtual IPv4 Address for a Subordinate Group(VRRP) 235
 - Configuring a Secondary Virtual IPv4 address for the Subordinate Group 236
- MIB support for VRRP 237
 - Configuring SNMP server notifications for VRRP events 237
- VRRP Support on PWHE Interfaces 237
- Hot Restartability for VRRP 239
- Configuration Examples for VRRP Implementation on Cisco IOS XR Software 240
 - Configuring a VRRP Group: Example 240
 - Clearing VRRP Statistics: Example 241
- Additional References 241

CHAPTER 13**Configuring Transports 245**

- Prerequisites for Configuring NSR, SCTP, TCP, UDP, and RAW Transports 245
- Information About Configuring NSR, SCTP, TCP, UDP, and RAW Transports 246
 - NSR Overview 246

| | |
|--|-----|
| SCTP Overview | 246 |
| TCP Overview | 246 |
| UDP Overview | 247 |
| How to Configure Failover as a Recovery Action for NSR | 247 |
| Configuring Failover as a Recovery Action for NSR | 247 |
| Additional References | 248 |



Preface



Note This product has reached end-of-life status. For more information, see the [End-of-Life and End-of-Sale Notices](#).

The *IP Addresses and Services Configuration Guide for Cisco CRS Routers* preface contains these sections:

- [Changes to This Document, on page xv](#)
- [Communications, Services, and Additional Information, on page xv](#)

Changes to This Document

Table 1: Changes to This Document

| Date | Change Summary |
|-------------|-----------------------------------|
| August 2020 | Initial release of this document. |

Communications, Services, and Additional Information

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

Cisco Bug Search Tool

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



CHAPTER 1

New and Changed IP Addresses and Services Features

This table summarizes the new and changed feature information for the *IP Addresses and Services Configuration Guide for Cisco CRS Routers*, and tells you where they are documented.

- [IP Addresses and Services Features Added or Modified in IOS XR Release 6.7.x, on page 1](#)

IP Addresses and Services Features Added or Modified in IOS XR Release 6.7.x

This section describes the new and changed IP addresses features for Cisco IOS XR.

IP Addresses Features Added or Modified in IOS XR Release 6.7.x

Table 2: New and Changed Features

| Feature | Description | Changed in Release | Where Documented |
|-----------------------------|----------------|--------------------|------------------|
| No new features introduced. | Not applicable | Not applicable | Not applicable |



CHAPTER 2

Implementing Network Stack IPv4 and IPv6

The Network Stack IPv4 and IPv6 features are used to configure and monitor Internet Protocol Version 4 (IPv4) and Internet Protocol Version 6 (IPv6).

This module describes the new and revised tasks you need to implement Network Stack IPv4 and IPv6 on your Cisco IOS XR network.



Note For a complete description of the Network Stack IPv4 and IPv6 commands, refer to the *Network Stack IPv4 and IPv6 Commands* module of the *IP Addresses and Services Command Reference for Cisco CRS Routers*.

Feature History for Implementing Network Stack IPv4 and IPv6

| Release | Modification |
|---------------|---|
| Release 2.0 | This feature was introduced. |
| Release 3.8.0 | The Route-Tag Support for Connected Routes feature was added. |
| Release 3.9.0 | GRE for IPv4/ v6 feature was added. |
| Release 5.3.1 | IPv6 Virtual Addresses feature was added. |

- [Prerequisites for Implementing Network Stack IPv4 and IPv6](#), on page 4
- [Restrictions for Implementing Network Stack IPv4 and IPv6](#), on page 4
- [Information About Implementing Network Stack IPv4 and IPv6](#), on page 4
- [How to Implement Network Stack IPv4 and IPv6](#), on page 20
- [Generic Routing Encapsulation](#), on page 32
- [Configuration Examples for Implementing Network Stack IPv4 and IPv6](#), on page 33
- [Additional References](#), on page 33

Prerequisites for Implementing Network Stack IPv4 and IPv6

You must be in a user group associated with a task group that includes the proper task IDs. The command reference guides include the task IDs required for each command. If you suspect user group assignment is preventing you from using a command, contact your AAA administrator for assistance.

Restrictions for Implementing Network Stack IPv4 and IPv6

In any Cisco IOS XR software release with IPv6 support, multiple IPv6 global addresses can be configured on an interface. However, multiple IPv6 link-local addresses on an interface are not supported.

Information About Implementing Network Stack IPv4 and IPv6

To implement Network Stack IPv4 and IPv6, you need to understand the following concepts:

Network Stack IPv4 and IPv6 Exceptions

The Network Stack feature in the Cisco IOS XR software has the following exceptions:

- In Cisco IOS XR software, the **clear ipv6 neighbors** and **show ipv6 neighbors** commands include the **location node-id** keyword. If a location is specified, only the neighbor entries in the specified location are displayed.
- The **ipv6 nd scavenge-timeout** command sets the lifetime for neighbor entries in the stale state. When the scavenge-timer for a neighbor entry expires, the entry is cleared.
- In Cisco IOS XR software, the **show ipv4 interface** and **show ipv6 interface** commands include the **location node-id** keyword. If a location is specified, only the interface entries in the specified location are displayed.
- Cisco IOS XR software allows conflicting IP address entries at the time of configuration. If an IP address conflict exists between two interfaces that are active, Cisco IOS XR software brings down the interface according to the configured conflict policy, the default policy being to bring down the higher interface instance. For example, if GigabitEthernet 0/1/0/1 conflicts with GigabitEthernet 0/2/0/1, then the IPv4 protocol on GigabitEthernet 0/2/0/1 is brought down and IPv4 remains active on GigabitEthernet 0/1/0/1.

IPv4 and IPv6 Functionality

When Cisco IOS XR software is configured with both an IPv4 and an IPv6 address, the interface can send and receive data on both IPv4 and IPv6 networks.

The architecture of IPv6 has been designed to allow existing IPv4 users to make the transition easily to IPv6 while providing services such as end-to-end security, quality of service (QoS), and globally unique addresses. The larger IPv6 address space allows networks to scale and provide global reachability. The simplified IPv6 packet header format handles packets more efficiently. IPv6 prefix aggregation, simplified network renumbering, and IPv6 site multihoming capabilities provide an IPv6 addressing hierarchy that allows for more efficient

routing. IPv6 supports widely deployed routing protocols such as Intermediate System-to-Intermediate System (IS-IS), Open Shortest Path First (OSPF), and multiprotocol Border Gateway Protocol (BGP).

The IPv6 neighbor discovery (nd) process uses Internet Control Message Protocol (ICMP) messages and solicited-node multicast addresses to determine the link-layer address of a neighbor on the same network (local link), verify the reachability of a neighbor, and keep track of neighboring routers.

IPv6 for Cisco IOS XR Software

IPv6, formerly named IPng (next generation) is the latest version of the Internet Protocol (IP). IP is a packet-based protocol used to exchange data, voice, and video traffic over digital networks. IPv6 was proposed when it became clear that the 32-bit addressing scheme of IP version 4 (IPv4) was inadequate to meet the demands of Internet growth. After extensive discussion, it was decided to base IPng on IP but add a much larger address space and improvements such as a simplified main header and extension headers. IPv6 is described initially in RFC 2460, *Internet Protocol, Version 6 (IPv6) Specification* issued by the Internet Engineering Task Force (IETF). Further RFCs describe the architecture and services supported by IPv6.

Larger IPv6 Address Space

The primary motivation for IPv6 is the need to meet the anticipated future demand for globally unique IP addresses. Applications such as mobile Internet-enabled devices (such as personal digital assistants [PDAs], telephones, and cars), home-area networks (HANs), and wireless data services are driving the demand for globally unique IP addresses. IPv6 quadruples the number of network address bits from 32 bits (in IPv4) to 128 bits, which provides more than enough globally unique IP addresses for every networked device on the planet. By being globally unique, IPv6 addresses inherently enable global reachability and end-to-end security for networked devices, functionality that is crucial to the applications and services that are driving the demand for the addresses. Additionally, the flexibility of the IPv6 address space reduces the need for private addresses and the use of Network Address Translation (NAT); therefore, IPv6 enables new application protocols that do not require special processing by border routers at the edge of networks.

IPv6 Address Formats

IPv6 addresses are represented as a series of 16-bit hexadecimal fields separated by colons (:) in the format: x:x:x:x:x:x:x:x. Following are two examples of IPv6 addresses:

```
2001:0DB8:7654:3210:FEDC:BA98:7654:3210
```

```
2001:0DB8:0:0:8:800:200C:417A
```

It is common for IPv6 addresses to contain successive hexadecimal fields of zeros. To make IPv6 addresses less cumbersome, two colons (::) can be used to compress successive hexadecimal fields of zeros at the beginning, middle, or end of an IPv6 address. (The colons represent successive hexadecimal fields of zeros.) [Table 3: Compressed IPv6 Address Formats, on page 6](#) lists compressed IPv6 address formats.

A double colon may be used as part of the *ipv6-address* argument when consecutive 16-bit values are denoted as zero. You can configure multiple IPv6 addresses per interfaces, but only one link-local address.



Note Two colons (::) can be used only once in an IPv6 address to represent the longest successive hexadecimal fields of zeros.

The hexadecimal letters in IPv6 addresses are not case-sensitive.

Table 3: Compressed IPv6 Address Formats

| IPv6 Address Type | Preferred Format | Compressed Format |
|-------------------|-------------------------------|--------------------------|
| Unicast | 2001:0:0:0:0DB8:800:200C:417A | 1080::0DB8:800:200C:417A |
| Multicast | FF01:0:0:0:0:0:0:101 | FF01::101 |
| Loopback | 0:0:0:0:0:0:0:1 | ::1 |
| Unspecified | 0:0:0:0:0:0:0:0 | :: |

The loopback address listed in [Table 3: Compressed IPv6 Address Formats, on page 6](#) may be used by a node to send an IPv6 packet to itself. The loopback address in IPv6 functions the same as the loopback address in IPv4 (127.0.0.1).



Note The IPv6 loopback address cannot be assigned to a physical interface. A packet that has the IPv6 loopback address as its source or destination address must remain within the node that created the packet. IPv6 routers do not forward packets that have the IPv6 loopback address as their source or destination address.

The unspecified address listed in [Table 3: Compressed IPv6 Address Formats, on page 6](#) indicates the absence of an IPv6 address. For example, a newly initialized node on an IPv6 network may use the unspecified address as the source address in its packets until it receives its IPv6 address.



Note The IPv6 unspecified address cannot be assigned to an interface. The unspecified IPv6 addresses must not be used as destination addresses in IPv6 packets or the IPv6 routing header.

An IPv6 address prefix, in the format *ipv6-prefix/prefix-length*, can be used to represent bit-wise contiguous blocks of the entire address space. The *ipv6-prefix* argument must be in the form documented in RFC 2373, in which the address is specified in hexadecimal using 16-bit values between colons. The prefix length is a decimal value that indicates how many of the high-order contiguous bits of the address compose the prefix (the network portion of the address). For example, 2001:0DB8:8086:6502::/32 is a valid IPv6 prefix.

IPv6 Address Type: Unicast

An IPv6 unicast address is an identifier for a single interface, on a single node. A packet that is sent to a unicast address is delivered to the interface identified by that address. Cisco IOS XR software supports the following IPv6 unicast address types:

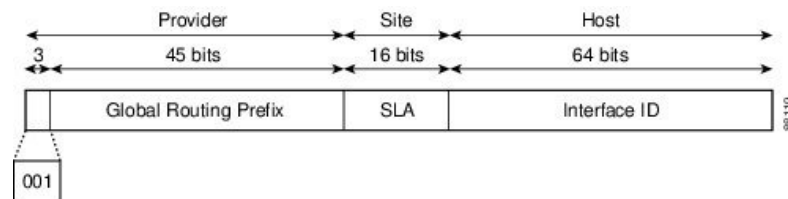
- Global aggregatable address
- Site-local address (proposal to remove by IETF)
- Link-local address
- IPv4-compatible IPv6 address

Aggregatable Global Address

An aggregatable global address is an IPv6 address from the aggregatable global unicast prefix. The structure of aggregatable global unicast addresses enables strict aggregation of routing prefixes that limits the number of routing table entries in the global routing table. Aggregatable global addresses are used on links that are aggregated upward through organizations, and eventually to the Internet service providers (ISPs).

Aggregatable global IPv6 addresses are defined by a global routing prefix, a subnet ID, and an interface ID. Except for addresses that start with binary 000, all global unicast addresses have a 64-bit interface ID. The current global unicast address allocation uses the range of addresses that start with binary value 001 (2000::/3). [Figure 1: Aggregatable Global Address Format, on page 7](#) shows the structure of an aggregatable global address.

Figure 1: Aggregatable Global Address Format



Addresses with a prefix of 2000::/3 (001) through E000::/3 (111) are required to have 64-bit interface identifiers in the extended universal identifier (EUI)-64 format. The Internet Assigned Numbers Authority (IANA) allocates the IPv6 address space in the range of 2000::/16 to regional registries.

The aggregatable global address typically consists of a 48-bit global routing prefix and a 16-bit subnet ID or Site-Level Aggregator (SLA). In the IPv6 aggregatable global unicast address format document (RFC 2374), the global routing prefix included two other hierarchically structured fields named Top-Level Aggregator (TLA) and Next-Level Aggregator (NLA). The IETF decided to remove the TLA and NLA fields from the RFCs, because these fields are policy-based. Some existing IPv6 networks deployed before the change might still be using networks based on the older architecture.

A 16-bit subnet field called the subnet ID could be used by individual organizations to create their own local addressing hierarchy and to identify subnets. A subnet ID is similar to a subnet in IPv4, except that an organization with an IPv6 subnet ID can support up to 65,535 individual subnets.

An interface ID is used to identify interfaces on a link. The interface ID must be unique to the link. It may also be unique over a broader scope. In many cases, an interface ID is the same as or based on the link-layer address of an interface. Interface IDs used in aggregatable global unicast and other IPv6 address types must be 64 bits long and constructed in the modified EUI-64 format.

Interface IDs are constructed in the modified EUI-64 format in one of the following ways:

- For all IEEE 802 interface types (for example, Ethernet interfaces and FDDI interfaces), the first three octets (24 bits) are taken from the Organizationally Unique Identifier (OUI) of the 48-bit link-layer address (MAC address) of the interface, the fourth and fifth octets (16 bits) are a fixed hexadecimal value of FFFE, and the last three octets (24 bits) are taken from the last three octets of the MAC address. The construction of the interface ID is completed by setting the Universal/Local (U/L) bit—the seventh bit of the first octet—to a value of 0 or 1. A value of 0 indicates a locally administered identifier; a value of 1 indicates a globally unique IPv6 interface identifier.
- For all other interface types (for example, serial, loopback, ATM, Frame Relay, and tunnel interface types—except tunnel interfaces used with IPv6 overlay tunnels), the interface ID is constructed in the same way as the interface ID for IEEE 802 interface types; however, the first MAC address from the

pool of MAC addresses in the router is used to construct the identifier (because the interface does not have a MAC address).

- For tunnel interface types that are used with IPv6 overlay tunnels, the interface ID is the IPv4 address assigned to the tunnel interface with all zeros in the high-order 32 bits of the identifier.



Note For interfaces using Point-to-Point Protocol (PPP), given that the interfaces at both ends of the connection might have the same MAC address, the interface identifiers used at both ends of the connection are negotiated (picked randomly and, if necessary, reconstructed) until both identifiers are unique. The first MAC address in the router is used to construct the identifier for interfaces using PPP.

If no IEEE 802 interface types are in the router, link-local IPv6 addresses are generated on the interfaces in the router in the following sequence:

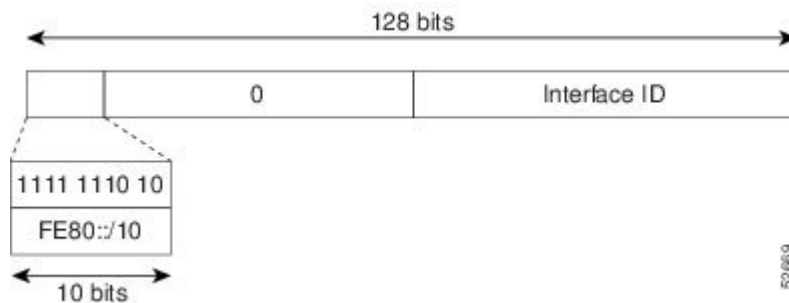
1. The router is queried for MAC addresses (from the pool of MAC addresses in the router).
2. If no MAC address is available, the serial number of the Route Processor (RP) or line card (LC) is used to form the link-local address.

Link-Local 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 neighbor discovery protocol and 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. [Figure 2: Link-Local Address Format, on page 8](#) shows the structure of a link-local address.

IPv6 routers must not forward packets that have link-local source or destination addresses to other links.

Figure 2: Link-Local Address Format

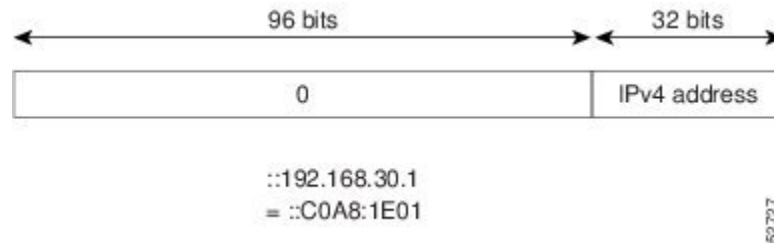


IPv4-Compatible IPv6 Address

An IPv4-compatible IPv6 address is an IPv6 unicast address that has zeros in the high-order 96 bits of the address and an IPv4 address in the low-order 32 bits of the address. The format of an IPv4-compatible IPv6 address is 0:0:0:0:0:A.B.C.D or ::A.B.C.D. The entire 128-bit IPv4-compatible IPv6 address is used as the IPv6 address of a node and the IPv4 address embedded in the low-order 32 bits is used as the IPv4 address.

of the node. IPv4-compatible IPv6 addresses are assigned to nodes that support both the IPv4 and IPv6 protocol stacks and are used in automatic tunnels. [Figure 3: IPv4-Compatible IPv6 Address Format, on page 9](#) shows the structure of an IPv4-compatible IPv6 address and a few acceptable formats for the address.

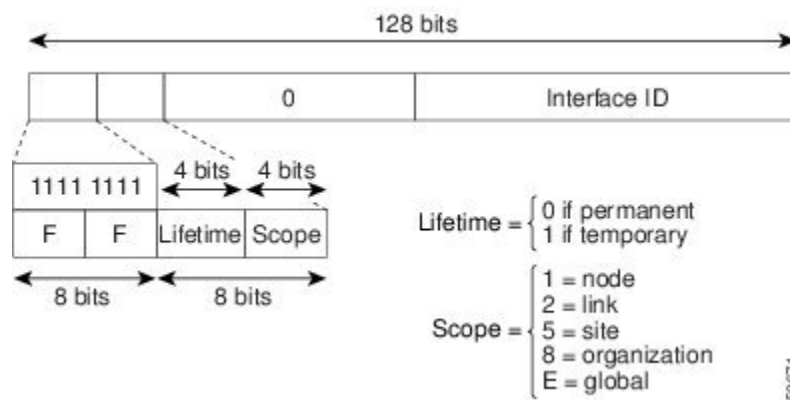
Figure 3: IPv4-Compatible IPv6 Address Format



IPv6 Address Type: Multicast

An IPv6 multicast address is an IPv6 address that has a prefix of FF00::/8 (1111 1111). An IPv6 multicast address is an identifier for a set of interfaces that typically belong to different nodes. A packet sent to a multicast address is delivered to all interfaces identified by the multicast address. The second octet following the prefix defines the lifetime and scope of the multicast address. A permanent multicast address has a lifetime parameter equal to 0; a temporary multicast address has a lifetime parameter equal to 1. A multicast address that has the scope of a node, link, site, or organization, or a global scope has a scope parameter of 1, 2, 5, 8, or E, respectively. For example, a multicast address with the prefix FF02::/16 is a permanent multicast address with a link scope. [Figure 4: IPv6 Multicast Address Format, on page 9](#) shows the format of the IPv6 multicast address.

Figure 4: IPv6 Multicast Address Format



IPv6 nodes (hosts and routers) are required to join (receive packets destined for) the following multicast groups:

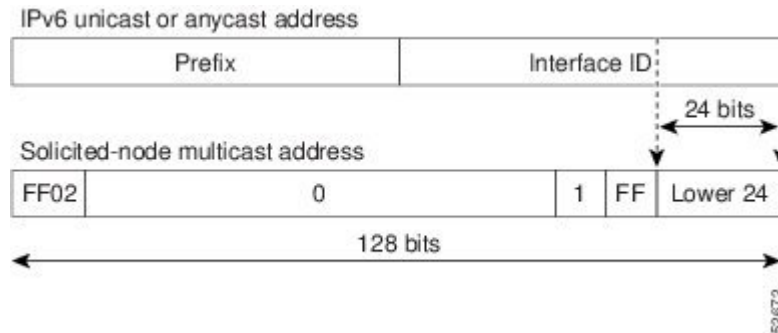
- All-nodes multicast group FF02:0:0:0:0:0:0:1 (scope is link-local)
- Solicited-node multicast group FF02:0:0:0:0:1:FF00:0000/104 for each of its assigned unicast and anycast addresses

IPv6 routers must also join the all-routers multicast group FF02:0:0:0:0:0:0:2 (scope is link-local).

The solicited-node multicast address is a multicast group that corresponds to an IPv6 unicast or anycast address. IPv6 nodes must join the associated solicited-node multicast group for every unicast and anycast address to

which it is assigned. The IPv6 solicited-node multicast address has the prefix FF02:0:0:0:0:1:FF00:0000/104 concatenated with the 24 low-order bits of a corresponding IPv6 unicast address. (See [Figure 5: IPv6 Solicited-Node Multicast Address Format, on page 10](#).) For example, the solicited-node multicast address corresponding to the IPv6 address 2037::01:800:200E:8C6C is FF02::1:FF0E:8C6C. Solicited-node addresses are used in neighbor solicitation messages.

Figure 5: IPv6 Solicited-Node Multicast Address Format



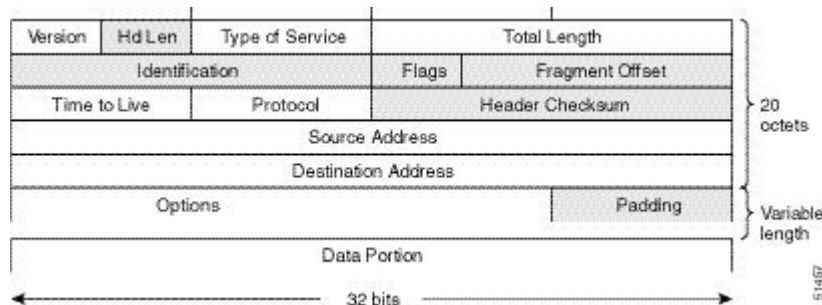
Note There are no broadcast addresses in IPv6. IPv6 multicast addresses are used instead of broadcast addresses.

For further information on IPv6 multicast, refer to the *Implementing Multicast* module in the *Multicast Configuration Guide for Cisco CRS Routers*.

Simplified IPv6 Packet Header

The basic IPv4 packet header has 12 fields with a total size of 20 octets (160 bits). The 12 fields may be followed by an Options field, which is followed by a data portion that is usually the transport-layer packet. The variable length of the Options field adds to the total size of the IPv4 packet header. The shaded fields of the IPv4 packet header are not included in the IPv6 packet header. (See [Figure 6: IPv4 Packet Header Format, on page 10](#))

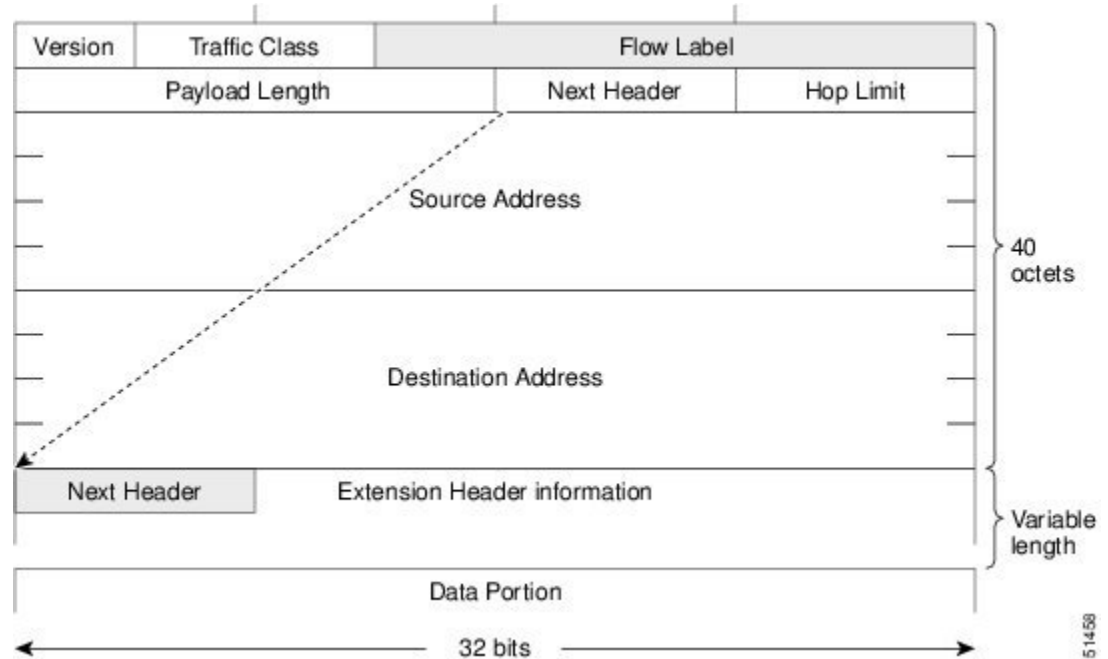
Figure 6: IPv4 Packet Header Format



The basic IPv6 packet header has 8 fields with a total size of 40 octets (320 bits). (See [Figure 7: IPv6 Packet Header Format, on page 11](#).) Fields were removed from the IPv6 header because, in IPv6, fragmentation is not handled by routers and checksums at the network layer are not used. Instead, fragmentation in IPv6 is handled by the source of a packet and checksums at the data link layer and transport layer are used. (In IPv4, the User Datagram Protocol (UDP) transport layer uses an optional checksum. In IPv6, use of the UDP

checksum is required to check the integrity of the inner packet.) Additionally, the basic IPv6 packet header and Options field are aligned to 64 bits, which can facilitate the processing of IPv6 packets.

Figure 7: IPv6 Packet Header Format



This table lists the fields in the basic IPv6 packet header.

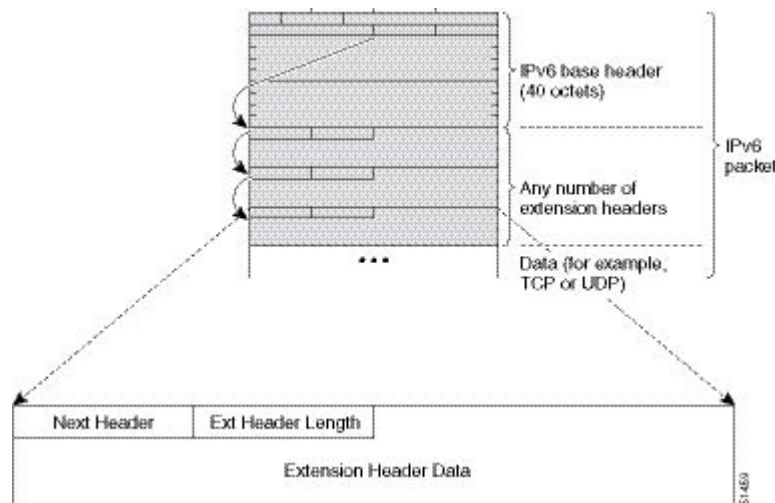
Table 4: Basic IPv6 Packet Header Fields

| Field | Description |
|----------------|---|
| Version | Similar to the Version field in the IPv4 packet header, except that the field lists number 6 for IPv6 instead of number 4 for IPv4. |
| Traffic Class | Similar to the Type of Service field in the IPv4 packet header. The Traffic Class field tags packets with a traffic class that is used in differentiated services. |
| Flow Label | A new field in the IPv6 packet header. The Flow Label field tags packets with a specific flow that differentiates the packets at the network layer. |
| Payload Length | Similar to the Total Length field in the IPv4 packet header. The Payload Length field indicates the total length of the data portion of the packet. |
| Next Header | Similar to the Protocol field in the IPv4 packet header. The value of the Next Header field determines the type of information following the basic IPv6 header. The type of information following the basic IPv6 header can be a transport-layer packet, for example, a TCP or UDP packet, or an Extension Header, as shown in Figure 8: IPv6 Extension Header Format, on page 12 . |

| Field | Description |
|---------------------|---|
| Hop Limit | Similar to the Time to Live field in the IPv4 packet header. The value of the Hop Limit field specifies the maximum number of routers that an IPv6 packet can pass through before the packet is considered invalid. Each router decrements the value by one. Because no checksum is in the IPv6 header, the router can decrement the value without needing to recalculate the checksum, which saves processing resources. |
| Source Address | Similar to the Source Address field in the IPv4 packet header, except that the field contains a 128-bit source address for IPv6 instead of a 32-bit source address for IPv4. |
| Destination Address | Similar to the Destination Address field in the IPv4 packet header, except that the field contains a 128-bit destination address for IPv6 instead of a 32-bit destination address for IPv4. |

Following the eight fields of the basic IPv6 packet header are optional extension headers and the data portion of the packet. If present, each extension header is aligned to 64 bits. There is no fixed number of extension headers in an IPv6 packet. Together, the extension headers form a chain of headers. Each extension header is identified by the Next Header field of the previous header. Typically, the final extension header has a Next Header field of a transport-layer protocol, such as TCP or UDP. [Figure 8: IPv6 Extension Header Format](#), on [page 12](#) shows the IPv6 extension header format.

Figure 8: IPv6 Extension Header Format



This table lists the extension header types and their Next Header field values.

Table 5: IPv6 Extension Header Types

| Header Type | Next Header Value | Description |
|---------------------------|-------------------|--|
| Hop-by-hop options header | 0 | This header is processed by all hops in the path of a packet. When present, the hop-by-hop options header always follows immediately after the basic IPv6 packet header. |

| Header Type | Next Header Value | Description |
|--------------------------------------|---------------------|--|
| Destination options header | 60 | The destination options header can follow any hop-by-hop options header, in which case the destination options header is processed at the final destination and also at each visited address specified by a routing header. Alternatively, the destination options header can follow any Encapsulating Security Payload (ESP) header, in which case the destination options header is processed only at the final destination. |
| Routing header | 43 | The routing header is used for source routing. |
| Fragment header | 44 | The fragment header is used when a source must fragment a packet that is larger than the maximum transmission unit (MTU) for the path between itself and a destination. The Fragment header is used in each fragmented packet. |
| Authentication header and ESP header | 51 50 | The Authentication header and the ESP header are used within IP Security Protocol (IPSec) to provide authentication, integrity, and confidentiality of a packet. These headers are identical for both IPv4 and IPv6. |
| Upper-layer header | 6 (TCP) 17 (UDP) | The upper-layer (transport) headers are the typical headers used inside a packet to transport the data. The two main transport protocols are TCP and UDP. |
| Mobility header | To be done by IANA | Extension headers used by mobile nodes, correspondent nodes, and home agents in all messaging related to the creation and management of bindings. |

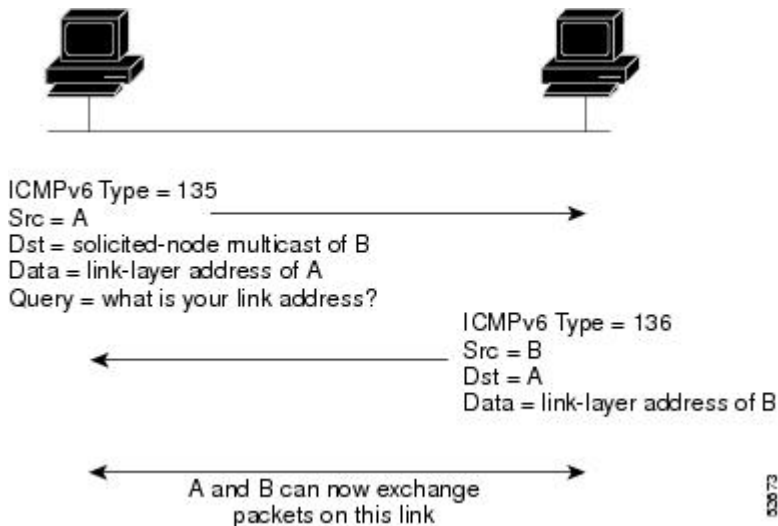
IPv6 Neighbor Discovery

The IPv6 neighbor discovery process uses ICMP messages and solicited-node multicast addresses to determine the link-layer address of a neighbor on the same network (local link), verify the reachability of a neighbor, and keep track of neighboring routers.

IPv6 Neighbor Solicitation Message

A value of 135 in the Type field of the ICMP packet header identifies a neighbor solicitation message. Neighbor solicitation messages are sent on the local link when a node wants to determine the link-layer address of another node on the same local link. (See [Figure 9: IPv6 Neighbor Discovery—Neighbor Solicitation Message, on page 14](#).) When a node wants to determine the link-layer address of another node, the source address in a neighbor solicitation message is the IPv6 address of the node sending the neighbor solicitation message. The destination address in the neighbor solicitation message is the solicited-node multicast address that corresponds to the IPv6 address of the destination node. The neighbor solicitation message also includes the link-layer address of the source node.

Figure 9: IPv6 Neighbor Discovery—Neighbor Solicitation Message



After receiving the neighbor solicitation message, the destination node replies by sending a neighbor advertisement message, which has a value of 136 in the Type field of the ICMP packet header, on the local link. The source address in the neighbor advertisement message is the IPv6 address of the node (more specifically, the IPv6 address of the node interface) sending the neighbor advertisement message. The destination address in the neighbor advertisement message is the IPv6 address of the node that sent the neighbor solicitation message. The data portion of the neighbor advertisement message includes the link-layer address of the node sending the neighbor advertisement message.

After the source node receives the neighbor advertisement, the source node and destination node can communicate.

Neighbor solicitation messages are also used to verify the reachability of a neighbor after the link-layer address of a neighbor is identified. When a node wants to verify the reachability of a neighbor, the destination address in a neighbor solicitation message is the unicast address of the neighbor.

Neighbor advertisement messages are also sent when there is a change in the link-layer address of a node on a local link. When there is such a change, the destination address for the neighbor advertisement is the all-nodes multicast address.

Neighbor solicitation messages are also used to verify the reachability of a neighbor after the link-layer address of a neighbor is identified. Neighbor unreachability detection identifies the failure of a neighbor or the failure of the forward path to the neighbor, and is used for all paths between hosts and neighboring nodes (hosts or routers). Neighbor unreachability detection is performed for neighbors to which only unicast packets are being sent and is not performed for neighbors to which multicast packets are being sent.

A neighbor is considered reachable when a positive acknowledgment is returned from the neighbor (indicating that packets previously sent to the neighbor have been received and processed). A positive acknowledgment—from an upper-layer protocol (such as TCP)—indicates that a connection is making forward progress (reaching its destination) or that a neighbor advertisement message in response to a neighbor solicitation message has been received. If packets are reaching the peer, they are also reaching the next-hop neighbor of the source. Therefore, forward progress is also a confirmation that the next-hop neighbor is reachable.

For destinations that are not on the local link, forward progress implies that the first-hop router is reachable. When acknowledgments from an upper-layer protocol are not available, a node probes the neighbor using unicast neighbor solicitation messages to verify that the forward path is still working. The return of a solicited

neighbor advertisement message from the neighbor is a positive acknowledgment that the forward path is still working. (Neighbor advertisement messages that have the solicited flag set to a value of 1 are sent only in response to a neighbor solicitation message.) Unsolicited messages confirm only the one-way path from the source to the destination node; solicited neighbor advertisement messages indicate that a path is working in both directions.



Note A neighbor advertisement message that has the solicited flag set to a value of 0 must not be considered as a positive acknowledgment that the forward path is still working.

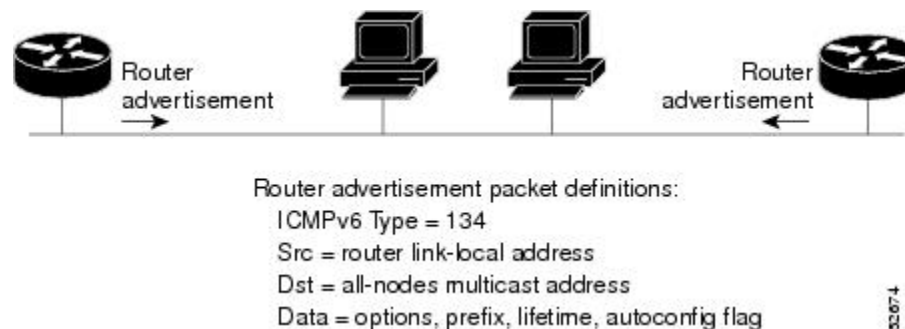
Neighbor solicitation messages are also used in the stateless autoconfiguration process to verify the uniqueness of unicast IPv6 addresses before the addresses are assigned to an interface. Duplicate address detection is performed first on a new, link-local IPv6 address before the address is assigned to an interface. (The new address remains in a tentative state while duplicate address detection is performed.) Specifically, a node sends a neighbor solicitation message with an unspecified source address and a tentative link-local address in the body of the message. If another node is already using that address, the node returns a neighbor advertisement message that contains the tentative link-local address. If another node is simultaneously verifying the uniqueness of the same address, that node also returns a neighbor solicitation message. If no neighbor advertisement messages are received in response to the neighbor solicitation message and no neighbor solicitation messages are received from other nodes that are attempting to verify the same tentative address, the node that sent the original neighbor solicitation message considers the tentative link-local address to be unique and assigns the address to the interface.

Every IPv6 unicast address (global or link-local) must be checked for uniqueness on the link; however, until the uniqueness of the link-local address is verified, duplicate address detection is not performed on any other IPv6 addresses associated with the link-local address. The Cisco implementation of duplicate address detection in the Cisco IOS XR software does not check the uniqueness of anycast or global addresses that are generated from 64-bit interface identifiers.

IPv6 Router Advertisement Message

Router advertisement (RA) messages, which have a value of 134 in the Type field of the ICMP packet header, are periodically sent out each configured interface of an IPv6 router. The router advertisement messages are sent to the all-nodes multicast address. (See [Figure 10: IPv6 Neighbor Discovery—Router Advertisement Message, on page 15.](#))

Figure 10: IPv6 Neighbor Discovery—Router Advertisement Message



Router advertisement messages typically include the following information:

- One or more onlink IPv6 prefixes that nodes on the local link can use to automatically configure their IPv6 addresses

- Lifetime information for each prefix included in the advertisement
- Sets of flags that indicate the type of autoconfiguration (stateless or statefull) that can be completed
- Default router information (whether the router sending the advertisement should be used as a default router and, if so, the amount of time, in seconds, that the router should be used as a default router)
- Additional information for hosts, such as the hop limit and MTU a host should use in packets that it originates

Router advertisements are also sent in response to router solicitation messages. 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 router advertisement message. Given that router solicitation messages are usually sent by hosts at system startup (the host does not have a configured unicast address), the source address in router solicitation messages is usually the unspecified IPv6 address (0:0:0:0:0:0:0:0). If the host has a configured unicast address, the unicast address of the interface sending the router solicitation message is used as the source address in the message. The destination address in router solicitation messages is the all-routers multicast address with a scope of the link. When a router advertisement is sent in response to a router solicitation, the destination address in the router advertisement message is the unicast address of the source of the router solicitation message.

The following router advertisement message parameters can be configured:

- The time interval between periodic router advertisement messages
- The “router lifetime” value, which indicates the usefulness of a router as the default router (for use by all nodes on a given link)
- The network prefixes in use on a given link
- The time interval between neighbor solicitation message retransmissions (on a given link)
- The amount of time a node considers a neighbor reachable (for use by all nodes on a given link)

The configured parameters are specific to an interface. The sending of router advertisement messages (with default values) is automatically enabled on Ethernet and FDDI interfaces. For other interface types, the sending of router advertisement messages must be manually configured by using the **no ipv6 nd suppress-ra** command in interface configuration mode. The sending of router advertisement messages can be disabled on individual interfaces by using the **ipv6 nd suppress-ra** command in interface configuration mode.

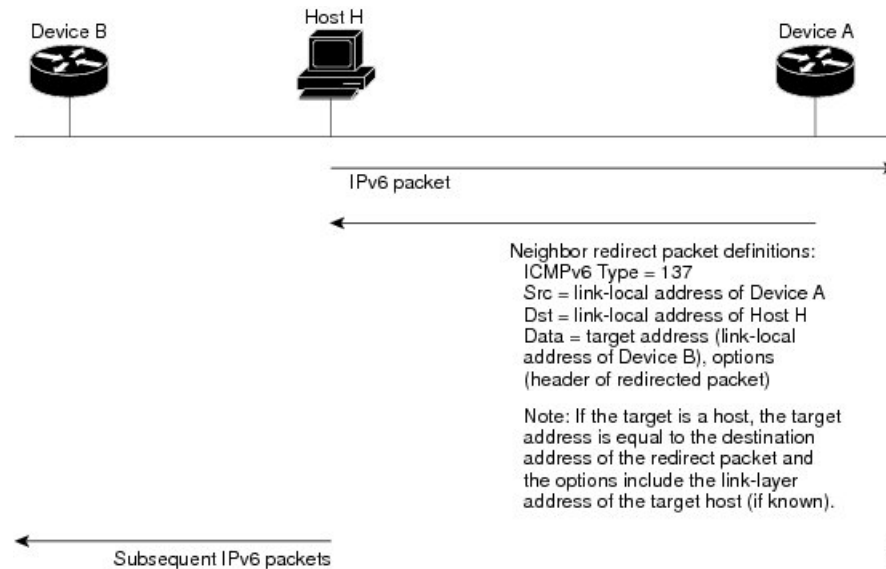


Note For stateless autoconfiguration to work properly, the advertised prefix length in router advertisement messages must always be 64 bits.

IPv6 Neighbor Redirect Message

A value of 137 in the Type field of the ICMP packet header identifies an IPv6 neighbor redirect message. Routers send neighbor redirect messages to inform hosts of better first-hop nodes on the path to a destination. (See [Figure 11: IPv6 Neighbor Discovery—Neighbor Redirect Message](#), on page 17.)

Figure 11: IPv6 Neighbor Discovery—Neighbor Redirect Message



Note A router must be able to determine the link-local address for each of its neighboring routers to ensure that the target address (the final destination) in a redirect message identifies the neighbor router by its link-local address. For static routing, the address of the next-hop router should be specified using the link-local address of the router; for dynamic routing, all IPv6 routing protocols must exchange the link-local addresses of neighboring routers.

After forwarding a packet, a router should send a redirect message to the source of the packet under the following circumstances:

- The destination address of the packet is not a multicast address.
- The packet was not addressed to the router.
- The packet is about to be sent out the interface on which it was received.
- The router determines that a better first-hop node for the packet resides on the same link as the source of the packet.
- The source address of the packet is a global IPv6 address of a neighbor on the same link, or a link-local address.

Use the **ipv6 icmp error-interval** global configuration command to limit the rate at which the router generates all IPv6 ICMP error messages, including neighbor redirect messages, which ultimately reduces link-layer congestion.



Note A router must not update its routing tables after receiving a neighbor redirect message, and hosts must not originate neighbor redirect messages.

ICMP for IPv6

Internet Control Message Protocol (ICMP) in IPv6 functions the same as ICMP in IPv4—ICMP generates error messages, such as ICMP destination unreachable messages and informational messages like ICMP echo request and reply messages. Additionally, ICMP packets in IPv6 are used in the IPv6 neighbor discovery process, path MTU discovery, and the Multicast Listener Discovery (MLD) protocol for IPv6. MLD is used by IPv6 routers to discover multicast listeners (nodes that want to receive multicast packets destined for specific multicast addresses) on directly attached links. MLD is based on version 2 of the Internet Group Management Protocol (IGMP) for IPv4.

A value of 58 in the Next Header field of the basic IPv6 packet header identifies an IPv6 ICMP packet. ICMP packets in IPv6 are like a transport-layer packet in the sense that the ICMP packet follows all the extension headers and is the last piece of information in the IPv6 packet. Within IPv6 ICMP packets, the ICMPv6 Type and ICMPv6 Code fields identify IPv6 ICMP packet specifics, such as the ICMP message type. The value in the Checksum field is derived (computed by the sender and checked by the receiver) from the fields in the IPv6 ICMP packet and the IPv6 pseudoheader. The ICMPv6 Data field contains error or diagnostic information relevant to IP packet processing.

Address Repository Manager

IPv4 and IPv6 Address Repository Manager (IPARM) enforces the uniqueness of global IP addresses configured in the system, and provides global IP address information dissemination to processes on route processors (RPs) and line cards (LCs) using the IP address consumer application program interfaces (APIs), which includes unnumbered interface information.

Address Conflict Resolution

There are two parts to conflict resolution; the conflict database and the conflict set definition.

Conflict Database

IPARM maintains a global conflict database. IP addresses that conflict with each other are maintained in lists called conflict sets. These conflict sets make up the global conflict database.

A set of IP addresses are said to be part of a conflict set if at least one prefix in the set conflicts with every other IP address belonging to the same set. For example, the following four addresses are part of a single conflict set.

address 1: 10.1.1.1/16

address 2: 10.2.1.1/16

address 3: 10.3.1.1/16

address 4: 10.4.1.1/8

When a conflicting IP address is added to a conflict set, an algorithm runs through the set to determine the highest precedence address within the set.

This conflict policy algorithm is deterministic, that is, the user can tell which addresses on the interface are enabled or disabled. The address on the interface that is enabled is declared as the highest precedence ip address for that conflict set.

The conflict policy algorithm determines the highest precedence ip address within the set.

Multiple IP Addresses

The IPARM conflict handling algorithm allows multiple IP addresses to be enabled within a set. Multiple addresses could potentially be highest precedence IP addresses.

```
interface GigabitEthernet 0/2/0/0: 10.1.1.1/16
```

```
interface GigabitEthernet 0/3/0/0: 10.1.1.2/8
```

```
interface GigabitEthernet 0/4/0/0: 10.2.1.1/16
```

The IP address on GigabitEthernet 0/2/0/0 is declared as highest precedence as per the lowest rack/slot policy and is enabled. However, because the address on interface GigabitEthernet 0/4/0/0 does not conflict with the current highest precedence IP address, the address on GigabitEthernet 0/4/0/0 is enabled as well.

Recursive Resolution of Conflict Sets

In the example below, the address on the interface in GigabitEthernet 0/2/0/0 has the highest precedence because it is the lowest rack/slot. However, now the addresses on GigabitEthernet 0/4/0/0 and GigabitEthernet 0/5/0/0 also do not conflict with the highest precedence IP addresses on GigabitEthernet 0/2/0/0. However, the addresses on GigabitEthernet 0/4/0/0 and GigabitEthernet 0/5/0/0 conflict with each other. The conflict resolution software tries to keep the interface that is enabled as the one that needs to stay enabled. If both interfaces are disabled, the software enables the address based on the current conflict policy. Because GigabitEthernet 0/4/0/0 is on a lower rack/slot, it is enabled.

```
interface GigabitEthernet 0/2/0/0: 10.1.1.1/16
```

```
interface GigabitEthernet 0/3/0/0: 10.1.1.2/8
```

```
interface GigabitEthernet 0/4/0/0: 10.2.1.1/16
```

```
interface GigabitEthernet 0/5/0/0: 10.2.1.2/16
```

Route-Tag Support for Connected Routes

The Route-Tag Support for Connected Routes feature that attaches a tag with all IPv4 and IPv6 addresses of an interface. The tag is propagated from the IPv4 and IPv6 management agents (MA) to the IPv4 and IPv6 address repository managers (ARM) to routing protocols, thus enabling the user to control the redistribution of connected routes by looking at the route tags, by using routing policy language (RPL) scripts. This prevents the redistribution of some interfaces, by checking for route tags in a route policy.

The route tag feature is already available for static routes and connected routes (interfaces) wherein the route tags are matched to policies and redistribution can be prevented.

SUMMARY STEPS

1. **configure**
2. **interface** *type interface-path-id*
3. Do one of the following:
 - **ipv4 address** *ipv4-address mask [secondary]*
 - **ipv6 address** *ipv6-prefix/prefix-length [eui-64]*
4. **route-tag** [*route-tag value*]
5. **commit**

DETAILED STEPS

| | Command or Action | Purpose |
|--------|---|---|
| Step 1 | configure | |
| Step 2 | interface <i>type interface-path-id</i> Example: RP/0/RP0/CPU0:router(config)# interface gigabitethernet 0/1/0/1 | Enters interface configuration mode. |
| Step 3 | Do one of the following: • ipv4 address <i>ipv4-address mask [secondary]</i> • ipv6 address <i>ipv6-prefix/prefix-length [eui-64]</i> Example: RP/0/RP0/CPU0:router(config-if)# ipv4 address 192.168.1.27 255.0.0.0 | Specifies a primary (or secondary) IPv4 address or an IPv6 address for an interface. |
| Step 4 | route-tag [<i>route-tag value</i>] Example: RP/0/RP0/CPU0:router(config-if)# ipv4 address 192.168.1.27 255.0.0.0 route-tag 100 | Specifies that the configured address has a route tag to be associated with it. The range for the route-tag value is 1 to 4294967295. |
| Step 5 | commit | |

How to Implement Network Stack IPv4 and IPv6

This section contains the following procedures:

Assigning IPv4 Addresses to Network Interfaces

This task assigns IPv4 addresses to individual network interfaces.

IPv4 Addresses

A basic and required task for configuring IP is to assign IPv4 addresses to network interfaces. Doing so enables the interfaces and allows communication with hosts on those interfaces using IPv4. An IP address identifies a location to which IP datagrams can be sent. An interface can have one primary IP address and multiple (up to 500) secondary addresses. Packets generated by the software always use the primary IPv4 address. Therefore, all networking devices on a segment should share the same primary network number.

Associated with this task are decisions about subnetting and masking the IP addresses. A mask identifies the bits that denote the network number in an IP address. When you use the mask to subnet a network, the mask is then referred to as a *subnet mask*.



Note Cisco supports only network masks that use contiguous bits that are flush left against the network field.

SUMMARY STEPS

1. **configure**
2. **interface** *type interface-path-id*
3. **ipv4 address** *ipv4-address mask* [**secondary**]
4. **commit**
5. show ipv4 interface

DETAILED STEPS

| | Command or Action | Purpose |
|---------------|--|--|
| Step 1 | configure | |
| Step 2 | interface <i>type interface-path-id</i> Example: RP/0/RP0/CPU0:router(config)# interface gigabitethernet 0/1/0/1 | Enters interface configuration mode. |
| Step 3 | ipv4 address <i>ipv4-address mask</i> [secondary] Example: RP/0/RP0/CPU0:router(config-if)# ipv4 address 192.168.1.27 255.0.0.0 RP/0/RP0/CPU0:router(config-if)# ipv4 address 192.168.1.27/8 | Specifies a primary or secondary IPv4 address for an interface. <ul style="list-style-type: none"> • The network mask can be a four-part dotted decimal address. For example, 255.0.0.0 indicates that each bit equal to 1 means the corresponding address bit belongs to the network address. • The network mask can be indicated as a slash (/) and a number- a prefix length. The prefix length is a decimal value that indicates how many of the high-order contiguous bits of the address comprise the prefix (the network portion of the address). A slash must precede the decimal value, and there is no space between the IP address and the slash. |
| Step 4 | commit | |
| Step 5 | show ipv4 interface Example: RP/0/RP0/CPU0:router# show ipv4 interface | (Optional) Displays the usability status of interfaces configured for IPv4. |

IPv4 Virtual Addresses

Configuring an IPv4 virtual address enables you to access the router from a single virtual address with a management network, without the prior knowledge of which route processor (RP) is active. An IPv4 virtual

address persists across RP failover situations. For this to happen, the virtual IPv4 address must share a common IPv4 subnet with a Management Ethernet interface on both RPs.

The **vrf** keyword supports virtual addresses on a per-VRF basis.

The **use-as-src-addr** keyword eliminates the need for configuring a loopback interface as the source interface (that is, update source) for management applications. When an update source is not configured, management applications allow the transport processes (TCP, UDP, raw_ip) to select a suitable source address. The transport processes, in turn, consult the FIB for selecting a suitable source address. If a Management Ethernet's IP address is selected as the source address and if the **use-as-src-addr** keyword is configured, then the transport substitutes the Management Ethernet's IP address with a relevant virtual IP address. This functionality works across RP switchovers. If the **use-as-src-addr** is not configured, then the source-address selected by transports can change after a failover and the NMS software may not be able to manage this situation.



Note Protocol configuration such as tacacs source-interface, snmp-server trap-source, ntp source, logging source-interface do not use the virtual management IP address as their source by default. Use the **ipv4 virtual address use-as-src-addr** command to ensure that the protocol uses the virtual IPv4 address as its source address. Alternatively, you can also configure a loopback address with the designated or desired IPv4 address and set that as the source for protocols such as TACACS+ via the **tacacs source-interface** command.

Configuring IPv6 Addressing

This task assigns IPv6 addresses to individual router interfaces and enable the forwarding of IPv6 traffic globally on the router. By default, IPv6 addresses are not configured.



Note The *ipv6-prefix* argument in the **ipv6 address** command must be in the form documented in RFC 2373 in which the address is specified in hexadecimal using 16-bit values between colons.

The **/prefix-length** argument in the **ipv6 address** command is a decimal value that indicates how many of the high-order contiguous bits of the address comprise the prefix (the network portion of the address) A slash must precede the decimal value.

The *ipv6-address* argument in the **ipv6 address link-local** command must be in the form documented in RFC 2373 where the address is specified in hexadecimal using 16-bit values between colons.

IPv6 Multicast Groups

An IPv6 address must be configured on an interface for the interface to forward IPv6 traffic. Configuring a global IPv6 address on an interface automatically configures a link-local address and activates IPv6 for that interface.

Additionally, the configured interface automatically joins the following required multicast groups for that link:

- Solicited-node multicast group FF02:0:0:0:1:FF00::/104 for each unicast address assigned to the interface
- All-nodes link-local multicast group FF02::1

- All-routers link-local multicast group FF02::2



Note The solicited-node multicast address is used in the neighbor discovery process.

SUMMARY STEPS

1. **configure**
2. **interface** *type interface-path-id*
3. Do one of the following:
 - **ipv6 address** *ipv6-prefix / prefix-length* [**eui-64**]
 - **ipv6 address** *ipv6-address* **link-local**
 - **ipv6 enable**
4. **commit**

DETAILED STEPS

| | Command or Action | Purpose |
|--------|--|--|
| Step 1 | configure | |
| Step 2 | interface <i>type interface-path-id</i> Example: <pre>RP/0/RP0/CPU0:router(config)# interface gigabitethernet 0/1/0/3</pre> | Enters interface configuration mode. |
| Step 3 | Do one of the following: <ul style="list-style-type: none"> • ipv6 address <i>ipv6-prefix / prefix-length</i> [eui-64] • ipv6 address <i>ipv6-address</i> link-local • ipv6 enable Example: <pre>RP/0/RP0/CPU0:router(config-if)# ipv6 address 2001:0DB8:0:1::/64 eui-64</pre> <pre>RP/0/RP0/CPU0:router(config-if)# ipv6 address 2001:0DB8:0:1::1/64</pre> or <pre>RP/0/RP0/CPU0:router(config-if)# ipv6 address FE80::260:3EFF:FE11:6770 link-local</pre> or <pre>RP/0/RP0/CPU0:router(config-if)# ipv6 enable</pre> | Specifies an IPv6 network assigned to the interface and enables IPv6 processing on the interface. or Automatically configures an IPv6 link-local address on the interface while also enabling the interface for IPv6 processing. The link-local address can be used only to communicate with nodes on the same link. <ul style="list-style-type: none"> • Specifying the ipv6 address eui-64 command configures site-local and global IPv6 addresses with an interface identifier (ID) in the low-order 64 bits of the IPv6 address. Only the 64-bit network prefix for the address needs to be specified; the last 64 bits are automatically computed from the interface ID. • Specifying the ipv6 address link-local command configures a link-local address on the interface that is used instead of the link-local address that is automatically configured when IPv6 is enabled on the interface. |

| | Command or Action | Purpose |
|--------|-------------------|---|
| | | or Enables IPv6 processing on an interface that has not been configured with an explicit IPv6 address. |
| Step 4 | commit | |

IPv6 Virtual Addresses

Configuring an IPv6 virtual address enables you to access the router from a single virtual address with a management network, without the prior knowledge of which route processor (RP) is active. An IPv6 virtual address persists across RP failover situations. For this to happen, the virtual IPv6 address must share a common IPv6 subnet with a Management Ethernet interface on both RPs.

The **vrf** keyword supports virtual addresses on a per-VRF basis.

The **use-as-src-addr** keyword eliminates the need for configuring a loopback interface as the source interface (that is, update source) for management applications. When an update source is not configured, management applications allow the transport processes (TCP, UDP, raw_ip) to select a suitable source address. The transport processes, in turn, consult the FIB for selecting a suitable source address. If a Management Ethernet's IP address is selected as the source address and if the **use-as-src-addr** keyword is configured, then the transport substitutes the Management Ethernet's IP address with a relevant virtual IP address. This functionality works across RP switchovers. If the **use-as-src-addr** is not configured, then the source-address selected by transports can change after a failover and the NMS software may not be able to manage this situation.



Note Protocol configuration such as tacacs source-interface, snmp-server trap-source, ntp source, logging source-interface do not use the virtual management IP address as their source by default. Use the **ipv6 virtual address use-as-src-addr** command to ensure that the protocol uses the virtual IPv6 address as its source address. Alternatively, you can also configure a loopback address with the designated or desired IPv6 address and set that as the source for protocols such as TACACS+ via the **tacacs source-interface** command.

Assigning Multiple IP Addresses to Network Interfaces

This task assigns multiple IP addresses to network interfaces.

Secondary IPv4 Addresses

The Cisco IOS XR software supports multiple IP addresses per interface.

You can specify a maximum of 500 secondary addresses. Secondary IP addresses can be used in a variety of situations. The following are the most common applications:

- There might not be enough host addresses for a particular network segment. For example, suppose your subnetting allows up to 254 hosts per logical subnet, but on one physical subnet you must have 300 host addresses. Using secondary IP addresses on the routers or access servers allows you to have two logical subnets using one physical subnet.

- Many older networks were built using Level 2 bridges, and were not subnetted. The judicious use of secondary addresses can aid in the transition to a subnetted, router-based network. Routers on an older, bridged segment can easily be made aware that many subnets are on that segment.
- Two subnets of a single network might otherwise be separated by another network. You can create a single network from subnets that are physically separated by another network by using a secondary address. In these instances, the first network is *extended*, or layered on top of the second network. Note that a subnet cannot appear on more than one active interface of the router at a time.



Note If any router on a network segment uses a secondary IPv4 address, all other routers on that same segment must also use a secondary address from the same network or subnet.



Caution Inconsistent use of secondary addresses on a network segment can quickly cause routing loops.

SUMMARY STEPS

1. **configure**
2. **interface** *type interface-path-id*
3. **ipv4 address** *ipv4-address mask [secondary]*
4. **commit**

DETAILED STEPS

| | Command or Action | Purpose |
|--------|--|--|
| Step 1 | configure | |
| Step 2 | interface <i>type interface-path-id</i> Example: RP/0/RP0/CPU0:router(config)# interface gigabitethernet 0/1/0/3 | Enters interface configuration mode. |
| Step 3 | ipv4 address <i>ipv4-address mask [secondary]</i> Example: RP/0/RP0/CPU0:router(config-if)# ipv4 address 192.168.1.27 255.255.255.0 secondary | Specifies that the configured address is a secondary IPv4 address. |
| Step 4 | commit | |

Configuring IPv4 and IPv6 Protocol Stacks

This task configures an interface in a Cisco networking device to support both the IPv4 and IPv6 protocol stacks.

When an interface in a Cisco networking device is configured with both an IPv4 and an IPv6 address, the interface forwards both IPv4 and IPv6 traffic—the interface can send and receive data on both IPv4 and IPv6 networks.

SUMMARY STEPS

1. **configure**
2. **interface** *type interface-path-id*
3. **ipv4 address** *ip-address mask [secondary]*
4. **ipv6 address** *ipv6-prefix/prefix-length [eui-64]*
5. **commit**

DETAILED STEPS

| | Command or Action | Purpose |
|---------------|---|---|
| Step 1 | configure | |
| Step 2 | interface <i>type interface-path-id</i> Example: RP/0/RP0/CPU0:router(config)# interface gigabitethernet 0/1/0/1 | Specifies the interface type and number, and enters interface configuration mode. |
| Step 3 | ipv4 address <i>ip-address mask [secondary]</i> Example: RP/0/RP0/CPU0:router(config-if)# ipv4 address 192.168.99.1 255.255.255.0 | Specifies a primary or secondary IPv4 address for an interface. |
| Step 4 | ipv6 address <i>ipv6-prefix/prefix-length [eui-64]</i> Example: RP/0/RP0/CPU0:router(config-if)# ipv6 address 2001:0DB8:c18:1::3/64 | Specifies the IPv6 address assigned to the interface and enables IPv6 processing on the interface. <ul style="list-style-type: none"> • A slash mark (/) must precede the <i>prefix-length</i>, and there is no space between the <i>ipv6-prefix</i> and slash mark. |
| Step 5 | commit | |

Enabling IPv4 Processing on an Unnumbered Interface

This task enables IPv4 processing on an unnumbered interface.

IPv4 Processing on an Unnumbered Interface

This section describes the process of enabling an IPv4 point-to-point interface without assigning an explicit IP address to the interface. Whenever the unnumbered interface generates a packet (for example, for a routing update), it uses the address of the interface you specified as the source address of the IP packet. It also uses the specified interface address in determining which routing processes are sending updates over the unnumbered interface. Restrictions are as follows:

- Serial interfaces using High-Level Data Link Control (HDLC), PPP, and Frame Relay encapsulations can be unnumbered. Serial interfaces using Frame Relay encapsulation can also be unnumbered, but the interface must be a point-to-point subinterface.
- You cannot use the **ping** EXEC command to determine whether the interface is up, because the interface has no IP address. The Simple Network Management Protocol (SNMP) can be used to remotely monitor interface status.
- You cannot support IP security options on an unnumbered interface.
- If you have configured GRE tunnel as IPv4 unnumbered interface then you must also configure a static route. The tunnel cannot reach the peer address if there is no static route configured. Here is the configuration example:

```
interface Loopback 100
ipv4 address 192.0.2.1 255.255.255.252

interface tunnel-ip 100
ipv4 unnumbered Loopback 100
tunnel source 192.0.2.10
keepalive
tunnel destination 192.0.2.11

router static
address-family ipv4 unicast
  192.0.2.2/32 tunnel-ip 100
```

If you are configuring Intermediate System-to-Intermediate System (IS-IS) across a serial line, you should configure the serial interfaces as unnumbered, which allows you to conform with RFC 1195, which states that IP addresses are not required on each interface.

SUMMARY STEPS

1. **configure**
2. **interface** *type interface-path-id*
3. **ipv4 unnumbered** *interface-type interface-instance*
4. **commit**

DETAILED STEPS

| | Command or Action | Purpose |
|---------------|--|---|
| Step 1 | configure | |
| Step 2 | interface <i>type interface-path-id</i> Example: RP/0/RP0/CPU0:router(config)# interface gigabitethernet 0/1/0/1 | Enters interface configuration mode. |
| Step 3 | ipv4 unnumbered <i>interface-type interface-instance</i> Example: RP/0/RP0/CPU0:router(config-if)# ipv4 unnumbered loopback 5 | Enables IPv4 processing on a point-to-point interface without assigning an explicit IPv4 address to that interface. <ul style="list-style-type: none"> • The interface you specify must be the name of another interface in the router that has an IP address, not another unnumbered interface. |

| | Command or Action | Purpose |
|--------|-------------------|--|
| | | <ul style="list-style-type: none"> The interface you specify by the <i>interface-type</i> and <i>interface-instance</i> arguments must be enabled (listed as “up” in the show interfaces command display). |
| Step 4 | commit | |

Configuring ICMP Rate Limiting

This task explains how to configure IPv4 or IPv6 ICMP rate limiting.

IPv4 ICMP Rate Limiting

The IPv4 ICMP rate limiting feature limits the rate that IPv4 ICMP destination unreachable messages are generated. The Cisco IOS XR software maintains two timers: one for general destination unreachable messages and one for DF destination unreachable messages. Both share the same time limits and defaults. If the **DF** keyword is not configured, the **icmp ipv4 rate-limit unreachable** command sets the time values for DF destination unreachable messages. If the **DF** keyword is configured, its time values remain independent from those of general destination unreachable messages.

IPv6 ICMP Rate Limiting

The IPv6 ICMP rate limiting feature implements a token bucket algorithm for limiting the rate at which IPv6 ICMP error messages are sent out on the network. The initial implementation of IPv6 ICMP rate limiting defined a fixed interval between error messages, but some applications, such as traceroute, often require replies to a group of requests sent in rapid succession. The fixed interval between error messages is not flexible enough to work with applications such as traceroute and can cause the application to fail. Implementing a token bucket scheme allows a number of tokens—representing the ability to send one error message each—to be stored in a virtual bucket. The maximum number of tokens allowed in the bucket can be specified, and for every error message to be sent, one token is removed from the bucket. If a series of error messages is generated, error messages can be sent until the bucket is empty. When the bucket is empty of tokens, IPv6 ICMP error messages are not sent until a new token is placed in the bucket. The token bucket algorithm does not increase the average rate limiting time interval, and it is more flexible than the fixed time interval scheme.

SUMMARY STEPS

- configure**
- Do one of the following:
 - icmp ipv4 rate-limit unreachable [DF] milliseconds**
 - ipv6 icmp error-interval milliseconds [bucketsize]**
- commit**
- Do one of the following:
 - show ipv4 traffic [brief]**
 - show ipv6 traffic [brief]**

DETAILED STEPS

| | Command or Action | Purpose |
|--------|--|---|
| Step 1 | configure | |
| Step 2 | <p>Do one of the following:</p> <ul style="list-style-type: none"> • icmp ipv4 rate-limit unreachable [DF] milliseconds • ipv6 icmp error-interval milliseconds [bucketsize] <p>Example:</p> <pre>RP/0/RP0/CPU0:router(config)# icmp ipv4 rate-limit unreachable 1000</pre> <p>or</p> <pre>RP/0/RP0/CPU0:router(config)# ipv6 icmp error-interval 50 20</pre> | <p>Limits the rate that IPv4 ICMP destination unreachable messages are generated.</p> <ul style="list-style-type: none"> • The DF keyword limits the rate at which ICMP destination unreachable messages are sent when code 4 fragmentation is needed and Data Fragmentation (DF) is set, as specified in the IP header of the ICMP destination unreachable message. • The <i>milliseconds</i> argument specifies the time period between the sending of ICMP destination unreachable messages. <p>or</p> <p>Configures the interval and bucket size for IPv6 ICMP error messages.</p> <ul style="list-style-type: none"> • The <i>milliseconds</i> argument specifies the interval between tokens being added to the bucket. • The optional <i>bucketsize</i> argument defines the maximum number of tokens stored in the bucket. |
| Step 3 | commit | |
| Step 4 | <p>Do one of the following:</p> <ul style="list-style-type: none"> • show ipv4 traffic [brief] • show ipv6 traffic [brief] <p>Example:</p> <pre>RP/0/RP0/CPU0:router# show ipv4 traffic</pre> <p>or</p> <pre>RP/0/RP0/CPU0:router# show ipv6 traffic</pre> | <p>(Optional) Displays statistics about IPv4 traffic, including ICMP unreachable information.</p> <ul style="list-style-type: none"> • Use the brief keyword to display only IPv4 and ICMPv4 traffic statistics. <p>or</p> <p>(Optional) Displays statistics about IPv6 traffic, including IPv6 ICMP rate-limited counters.</p> <ul style="list-style-type: none"> • Use the brief keyword to display only IPv6 and ICMPv6 traffic statistics. |

Selecting Flexible Source IP

To allow for flexible source IP address selection in the Internet Control Message Protocol (ICMP) response packet in response to a failure, use the `icmp source` command in the Global Configuration mode.

SUMMARY STEPS

1. `configure`
2. `icmp [ipv4 | ipv6] source [vrf | rfc]`

3. commit**DETAILED STEPS**

| | Command or Action | Purpose |
|---------------|--|--|
| Step 1 | configure Example: RP/0/RP0/CPU0:router# configure | Enters global configuration mode. |
| Step 2 | icmp [ipv4 ipv6] source [vrf rfc] Example: RP/0/RP0/CPU0:router(config)# icmp ipv6 source vrf | Allows flexible source IP address corresponding to strict vrf in outgoing IPv6 ICMP packets. |
| Step 3 | commit | |

Configuring IPARM Conflict Resolution

This task sets the IP Address Repository Manager (IPARM) address conflict resolution parameters.

Static Policy Resolution

The static policy resolution configuration prevents new address configurations from affecting interfaces that are currently running.



Note When you configure duplicate IP addresses of interfaces on a device and also configure the command `ipv4 conflict-policy static`, the duplicate interface remains down. However, this configuration is applicable only on ethernet interfaces and not on Point-to-Point (PPP) interfaces and Cisco ASR 9000 Series SPA Interface Processor-700 (SIP-700).

SUMMARY STEPS

1. **configure**
2. **{ipv4 | ipv6} conflict-policy static**
3. **commit**

DETAILED STEPS

| | Command or Action | Purpose |
|---------------|-------------------|---------|
| Step 1 | configure | |

| | Command or Action | Purpose |
|---------------|---|---|
| Step 2 | {ipv4 ipv6} conflict-policy static Example: <pre>RP/0/RP0/CPU0:router(config)# ipv4 conflict-policy static</pre> or <pre>RP/0/RP0/CPU0:router(config)# ipv6 conflict-policy static</pre> | Sets the conflict policy to static, that is, prevents new interface addresses from affecting the currently running interface. |
| Step 3 | commit | |

Longest Prefix Address Conflict Resolution

This conflict resolution policy attempts to give highest precedence to the IP address that has the longest prefix length.

SUMMARY STEPS

1. **configure**
2. **{ ipv4 | ipv6 } conflict-policy longest-prefix**
3. **commit**

DETAILED STEPS

| | Command or Action | Purpose |
|---------------|---|---|
| Step 1 | configure | |
| Step 2 | { ipv4 ipv6 } conflict-policy longest-prefix Example: <pre>RP/0/RP0/CPU0:router(config)# ipv4 conflict-policy longest-prefix</pre> or <pre>RP/0/RP0/CPU0:router(config)# ipv6 conflict-policy longest-prefix</pre> | Sets the conflict policy to longest prefix, that is, all addresses within the conflict set that don't conflict with the longest prefix address of the currently running interface are allowed to run as well. |
| Step 3 | commit | |

Highest IP Address Conflict Resolution

This conflict resolution policy attempts to give highest precedence to the IP address that has the highest value.

SUMMARY STEPS

1. **configure**
2. **{ ipv4 | ipv6 } conflict-policy highest-ip**

3. commit

DETAILED STEPS

| | Command or Action | Purpose |
|--------|---|---|
| Step 1 | <code>configure</code> | |
| Step 2 | <p><code>{ ipv4 ipv6 } conflict-policy highest-ip</code></p> <p>Example:</p> <pre>RP/0/RP0/CPU0:router(config)# ipv4 conflict-policy highest-ip</pre> <p>or</p> <pre>RP/0/RP0/CPU0:router(config)# ipv6 conflict-policy highest-ip</pre> | Sets the conflict policy to the highest IP value, that is, the IP address with the highest value gets precedence. |
| Step 3 | <code>commit</code> | |

Generic Routing Encapsulation

The Generic Routing Encapsulation (GRE) tunneling protocol provides a simple, and generic approach for transporting packets of one protocol over another protocol by means of encapsulation. The packet that needs to be transported is first encapsulated in a GRE header, which is further encapsulated in another protocol like IPv4 or IPv6; and the packet is then forwarded to the destination.

A typical GRE-encapsulated packet includes:

- The delivery header
- The GRE header
- The payload packet

A payload packet is a packet that a system encapsulates and delivers to a destination. The payload is first encapsulated in a GRE packet. The resulting GRE packet can then be encapsulated in another outer protocol and then forwarded. This outer protocol is called the delivery protocol.



Note

- When IPv4 is being carried as the GRE payload, the Protocol Type field must be set to 0x800.

IPv6 as delivery and/or payload protocol is not included in the currently deployed versions of GRE.

IPv4 Forwarding over GRE Tunnels

Packets that are tunneled over GRE tunnels enter the router as normal IP packets. The packets are forwarded (routed) using the destination address of the IP packet. In the case of Equal Cost Multi Path (ECMP) scenarios, an output interface-adjacency is selected, based on a platform-specific L3 load balance (LB) hash. In the case of two stage forwarding platforms like CRS, the receive-adjacency of the selected output interface is used to

reach the packet to the egress line card that hosts the interface. If the selected output interface is a GRE interface, the ingress line card has to determine the actual physical interface that can be used to reach the GRE tunnel destination. To determine this, a second routing (forwarding) decision is made based on the GRE tunnel destination address, where the L3 load balance hash is re-applied to determine the physical interface. Once the egress physical interface is known, the packet is sent out of that interface, after it is first encapsulated with GRE header followed by the L2 rewrite header of the physical interface. After the GRE encapsulated packet reaches the remote tunnel endpoint router, the GRE packet is decapsulated. The destination address lookup of the outer IP header (this is the same as the tunnel destination address) will find a local address (receive) entry on the ingress line card.

The first step in GRE decapsulation is to qualify the tunnel endpoint, before admitting the GRE packet into the router, based on the combination of tunnel source (the same as source IP address of outer IP header) and tunnel destination (the same as destination IP address of outer IP header). If the received packet fails tunnel admittance qualification check, the packet is dropped by the decapsulation router. On successful tunnel admittance check, the decapsulation strips the outer IP and GRE header off the packet, then starts processing the inner payload packet as a regular packet.

When a tunnel endpoint decapsulates a GRE packet, which has an IPv4 packet as the payload, the destination address in the IPv4 payload packet header is used to forward the packet, and the TTL of the payload packet is decremented. Care should be taken when forwarding such a packet. If the destination address of the payload packet is the encapsulator of the packet (that is the other end of the tunnel), looping can occur. In such a case, the packet must be discarded.

Configuration Examples for Implementing Network Stack IPv4 and IPv6

This section provides the following configuration examples:

Assigning an Unnumbered Interface: Example

In the following example, the second interface (GigabitEthernet 0/1/0/1) is given the address of loopback interface 0. The loopback interface is unnumbered.

```
interface loopback 0
  ipv4 address 192.168.0.5 255.255.255.0
interface gigabitethernet 0/1/0/1
  ipv4 unnumbered loopback 0
```

Additional References

The following sections provide references related to implementing Network Stack IPv4 and IPv6.

Related Documents

| Related Topic | Document Title |
|--|--|
| Address resolution configuration tasks | <i>Configuring ARP</i> module in this publication. |

| Related Topic | Document Title |
|---|---|
| Mapping host names to IP addresses | <i>Host Services and Applications Commands</i> module in the <i>IP Addresses and Services Command Reference for Cisco CRS Routers</i> |
| Network stack IPv4 and IPv6 commands: complete command syntax, command modes, command history, defaults, usage guidelines, and examples | <i>Network Stack IPv4 and IPv6 Commands</i> section in the <i>IP Addresses and Services Command Reference for Cisco CRS Routers</i> |

Standards

| Standards | Title |
|---|-------|
| No new or modified standards are supported by this feature, and support for existing standards has not been modified by this feature. | — |

MIBs

| MIBs | MIBs Link |
|------|--|
| — | To locate and download MIBs using Cisco IOS XR software, use the Cisco MIB Locator found at the following URL and choose a platform under the Cisco Access Products menu: https://mibs.cloudapps.cisco.com/ITDIT/MIBS/servlet/index |

RFCs

| RFCs | Title |
|---|-------|
| No new or modified RFCs are supported by this feature, and support for existing RFCs has not been modified by this feature. | — |

Technical Assistance

| Description | Link |
|---|---|
| The Cisco Technical Support website contains thousands of pages of searchable technical content, including links to products, technologies, solutions, technical tips, and tools. Registered Cisco.com users can log in from this page to access even more content. | http://www.cisco.com/techsupport |



CHAPTER 3

Implementing ARP

Address resolution is the process of mapping network addresses to Media Access Control (MAC) addresses. This process is accomplished using the Address Resolution Protocol (ARP).



Note For a complete description of the ARP commands listed in this module, refer to the *IP Addresses and Services Command Reference for Cisco CRS Routers*.

Feature History for Configuring ARP

| Release | Modification |
|---------------|---|
| Release 2.0 | This feature was introduced. |
| Release 3.3.0 | The vrf keyword and <i>vrf-name</i> argument were added to arp commands. Merged the Setting ARP Encapsulation section with the Defining a Static ARP Cache Entry. |

- [Prerequisites for Configuring ARP](#) , on page 35
- [Restrictions for Configuring ARP](#) , on page 35
- [Information About Configuring ARP](#) , on page 36
- [How to Configure ARP](#) , on page 38
- [Additional References](#), on page 41

Prerequisites for Configuring ARP

- You must be in a user group associated with a task group that includes the proper task IDs. The command reference guides include the task IDs required for each command. If you suspect user group assignment is preventing you from using a command, contact your AAA administrator for assistance.

Restrictions for Configuring ARP

The following restrictions apply to configuring ARP :

- Reverse Address Resolution Protocol (RARP) is not supported.

- Due to a hardware limitation in the Ethernet SPA interfaces installed on all routers, when a packet contains a wrong destination address, the corresponding SPA drops the packet even if the ingress packet count is already incremented in the output of the **show interfaces** command.

Information About Configuring ARP

To configure ARP, you must understand the following concepts:

IP Addressing Overview

A device in the IP can have both a local address (which uniquely identifies the device on its local segment or LAN) and a network address (which identifies the network to which the device belongs). The local address is more properly known as a *data link address*, because it is contained in the data link layer (Layer 2 of the OSI model) part of the packet header and is read by data-link devices (bridges and all device interfaces, for example). The more technically inclined person will refer to local addresses as *MAC addresses*, because the MAC sublayer within the data link layer processes addresses for the layer.

To communicate with a device on Ethernet, for example, Cisco IOS XR software first must determine the 48-bit MAC or local data-link address of that device. The process of determining the local data-link address from an IP address is called *address resolution*.

Address Resolution on a Single LAN

The following process describes address resolution when the source and destination devices are attached to the same LAN:

1. End System A broadcasts an ARP request onto the LAN, attempting to learn the MAC address of End System B.
2. The broadcast is received and processed by all devices on the LAN, including End System B.
3. Only End System B replies to the ARP request. It sends an ARP reply containing its MAC address to End System A.
4. End System A receives the reply and saves the MAC address of End System B in its ARP cache. (The ARP cache is where network addresses are associated with MAC addresses.)
5. Whenever End System A needs to communicate with End System B, it checks the ARP cache, finds the MAC address of System B, and sends the frame directly, without needing to first use an ARP request.

Address Resolution When Interconnected by a Router

The following process describes address resolution when the source and destination devices are attached to different LANs that are interconnected by a router (only if proxy-arp is turned on):

1. End System Y broadcasts an ARP request onto the LAN, attempting to learn the MAC address of End System Z.
2. The broadcast is received and processed by all devices on the LAN, including Router X.
3. Router X checks its routing table and finds that End System Z is located on a different LAN.

4. Router X therefore acts as a proxy for End System Z. It replies to the ARP request from End System Y, sending an ARP reply containing its own MAC address as if it belonged to End System Z.
5. End System Y receives the ARP reply and saves the MAC address of Router X in its ARP cache, in the entry for End System Z.
6. When End System Y needs to communicate with End System Z, it checks the ARP cache, finds the MAC address of Router X, and sends the frame directly, without using ARP requests.
7. Router X receives the traffic from End System Y and forwards it to End System Z on the other LAN.

ARP and Proxy ARP

Two forms of address resolution are supported by Cisco IOS XR software: Address Resolution Protocol (ARP) and proxy ARP, as defined in RFC 826 and RFC 1027, respectively. Cisco IOS XR software also supports a form of ARP called local proxy ARP.

ARP is used to associate IP addresses with media or MAC addresses. Taking an IP address as input, ARP determines the associated media address. After a media or MAC address is determined, the IP address or media address association is stored in an ARP cache for rapid retrieval. Then the IP datagram is encapsulated in a link-layer frame and sent over the network.

When proxy ARP is disabled, the networking device responds to ARP requests received on an interface only if one of the following conditions is met:

- The target IP address in the ARP request is the same as the interface IP address on which the request is received.
- The target IP address in the ARP request has a statically configured ARP alias.

When proxy ARP is enabled, the networking device also responds to ARP requests that meet all the following conditions:

- The target IP address is not on the same physical network (LAN) on which the request is received.
- The networking device has one or more routes to the target IP address.
- All of the routes to the target IP address go through interfaces other than the one on which the request is received.

When local proxy ARP is enabled, the networking device responds to ARP requests that meet all the following conditions:

- The target IP address in the ARP request, the IP address of the ARP source, and the IP address of the interface on which the ARP request is received are on the same Layer 3 network.
- The next hop for the target IP address is through the same interface as the request is received.

Typically, local proxy ARP is used to resolve MAC addresses to IP addresses in the same Layer 3 network such as, private VLANs that are Layer 2-separated. Local proxy ARP supports all types of interfaces supported by ARP and unnumbered interfaces.

ARP Cache Entries

ARP establishes correspondences between network addresses (an IP address, for example) and Ethernet hardware addresses. A record of each correspondence is kept in a cache for a predetermined amount of time and then discarded.

You can also add a static (permanent) entry to the ARP cache that persists until expressly removed.

How to Configure ARP

This section contains instructions for the following tasks:

Defining a Static ARP Cache Entry

ARP and other address resolution protocols provide a dynamic mapping between IP addresses and media addresses. Because most hosts support dynamic address resolution, generally you need not to specify static ARP cache entries. If you must define them, you can do so globally. Performing this task installs a permanent entry in the ARP cache. Cisco IOS XR software uses this entry to translate 32-bit IP addresses into 48-bit hardware addresses.

Optionally, you can specify that the software responds to ARP requests as if it were the owner of the specified IP address by making an alias entry in the ARP cache.

SUMMARY STEPS

1. **configure**
2. Do one of the following:
 - `arp [vrf vrf-name] ip-address hardware-address encapsulation-type`
 - `arp [vrf vrf-name] ip-address hardware-address encapsulation-type alias`
3. **commit**

DETAILED STEPS

| | Command or Action | Purpose |
|--------|---|--|
| Step 1 | <code>configure</code> | |
| Step 2 | <p>Do one of the following:</p> <ul style="list-style-type: none"> • <code>arp [vrf vrf-name] ip-address hardware-address encapsulation-type</code> • <code>arp [vrf vrf-name] ip-address hardware-address encapsulation-type alias</code> <p>Example:</p> <pre>RP/0/RP0/CPU0:router(config)# arp 192.168.7.19 0800.0900.1834 arpa</pre> <p>or</p> | <p>Creates a static ARP cache entry associating the specified 32-bit IP address with the specified 48-bit hardware address.</p> <p>Note If an alias entry is created, then any interface to which the entry is attached will act as if it is the owner of the specified addresses, that is, it will respond to ARP request packets for this network layer address with the data link layer address in the entry.</p> |

| | Command or Action | Purpose |
|--------|---|---------|
| | RP/0/RP0/CPU0:router(config)# arp 192.168.7.19 0800.0900.1834 arpa alias | |
| Step 3 | commit | |

Enabling Proxy ARP

Cisco IOS XR software uses proxy ARP (as defined in RFC 1027) to help hosts with no knowledge of routing determine the media addresses of hosts on other networks or subnets. For example, if the router receives an ARP request for a host that is not on the same interface as the ARP request sender, and if the router has all of its routes to that host through other interfaces, then it generates a proxy ARP reply packet giving its own local data-link address. The host that sent the ARP request then sends its packets to the router, which forwards them to the intended host. Proxy ARP is disabled by default; this task describes how to enable proxy ARP if it has been disabled.

SUMMARY STEPS

1. **configure**
2. **interface** *type number*
3. proxy-arp
4. **commit**

DETAILED STEPS

| | Command or Action | Purpose |
|--------|---|--------------------------------------|
| Step 1 | configure | |
| Step 2 | interface <i>type number</i> Example: RP/0/RP0/CPU0:router(config)# interface MgmtEth 0/RP0/CPU0/0 | Enters interface configuration mode. |
| Step 3 | proxy-arp Example: RP/0/RP0/CPU0:router(config-if)# proxy-arp | Enables proxy ARP on the interface. |
| Step 4 | commit | |

Enabling Local Proxy ARP

Local proxy ARP is disabled by default; this task describes how to enable local proxy ARP.

SUMMARY STEPS

1. **configure**

2. `interface type number`
3. `local-proxy-arp`
4. `commit`

DETAILED STEPS

| | Command or Action | Purpose |
|--------|---|---|
| Step 1 | <code>configure</code> | |
| Step 2 | <p><code>interface type number</code></p> <p>Example:</p> <pre>RP/0/RP0/CPU0:router(config)# interface TenGigE 0/0/0/0</pre> | Enters interface configuration mode. |
| Step 3 | <p><code>local-proxy-arp</code></p> <p>Example:</p> <pre>RP/0/RP0/CPU0:router(config-if)# local-proxy-arp</pre> | Enables local proxy ARP on the interface. |
| Step 4 | <code>commit</code> | |

Configure Learning of Local ARP Entries

You can configure an interface or a sub-interface to learn only the ARP entries from its local subnet. Use the following procedure to configure local ARP learning on an interface.

1. Enter the interface configuration mode.

```
Router(config)# interface GigabitEthernet 0/0/0/1
```

2. Configure the IPv4/IPv6 address for the interface.

```
Router(config-if)# ipv4 address 12.1.3.4 255.255.255.0
```

3. Configure local ARP learning on the interface.

```
Router(config-if)# arp learning local
```

4. Enable the interface and commit your configuration.

```
Router(config-if)# no shut
Router(config-if)# commit
RP/0/0/CPU0:Dec 12 13:41:16.580 : ifmgr[397]: %PKT_INFRA-LINK-3-UPDOWN : interface
GigabitEthernet 0/0/0/1, changed state to Down
RP/0/0/CPU0:Dec 12 13:41:16.683 : ifmgr[397]: %PKT_INFRA-LINK-3-UPDOWN : interface
GigabitEthernet 0/0/0/1 changed state to Up
```

5. Confirm your configuration.

```
Router(config-if)# show running-configuration
..
Building configuration...
!! IOS XR Configuration 0.0.0
!! Last configuration change at Mon Dec 12 13:41:16 2016
!interface GigabitEthernet 0/0/0/1
 ipv4 address 12.1.3.4 255.255.255.0
```



```

    arp learning local
!

```

6. Verify if local ARP learning is working as configured on the interface.

```

Router(config-if)# do show arp idb gigabitEthernet 0/0/0/1 location 0/0/CPU0
Thu Dec 15 10:00:11.733 IST

```

```

GigabitEthernet 0/0/0/1 (0x00000040):
  IPv4 address 12.1.1.3.4, Vrf ID 0x60000000
  VRF Name default
  Dynamic learning: Local
  Dynamic entry timeout: 14400 secs
  Purge delay: off
  IPv4 caps added (state up)
  MPLS caps not added
  Interface not virtual, not client fwd ref,
  Proxy arp not configured, not enabled
  Local Proxy arp not configured
  Packet IO layer is NetIO
  Srg Role : DEFAULT
  Idb Flag : 2146444
  IDB is Complete

```

7. (Optional) You can monitor the ARP traffic on the interface.

```

Router(config-if)# do show arp idb gigabitEthernet 0/0/0/1 location 0/0/CPU0
Thu Dec 15 10:13:28.964 IST

```

```

ARP statistics:
  Recv: 0 requests, 0 replies
  Sent: 0 requests, 1 replies (0 proxy, 0 local proxy, 1 gratuitous)
  Subscriber Interface:
    0 requests rcvd, 0 replies sent, 0 gratuitous replies sent
  Resolve requests rcvd: 0
  Resolve requests dropped: 0
  Errors: 0 out of memory, 0 no buffers, 0 out of sunbet

```

```

ARP cache:
  Total ARP entries in cache: 1
  Dynamic: 0, Interface: 1, Standby: 0
  Alias: 0, Static: 0, DHCP: 0

  IP Packet drop count for GigabitEthernet0_0_0_1: 0

```

Additional References

The following sections provide references related to ARP.

Related Documents

| Related Topic | Document Title |
|--------------------------|--|
| ARP commands | <i>ARP Commands</i> module in <i>IP Addresses and Services Command Reference for Cisco CRS Routers</i> |
| Getting started material | <i>Cisco IOS XR Getting Started Guide for the Cisco CRS Router</i> |

| Related Topic | Document Title |
|---|---|
| QoS commands: complete command syntax, command modes, command history, defaults, usage guidelines, and examples | <i>Quality of Service Commands</i> module in <i>Modular QoS Command Reference for Cisco CRS Routers</i> |
| Class-based traffic shaping, traffic policing, low latency queuing, and MDDR | Configuring Modular Quality of Service Congestion Management module in <i>Modular QoS Configuration Guide for Cisco CRS Routers</i> |

Standards

| Standards | Title |
|---|-------|
| No new or modified standards are supported by this feature, and support for existing standards has not been modified by this feature. | — |

MIBs

| MIBs | MIBs Link |
|------|---|
| — | To locate and download MIBs, use the Cisco MIB Locator found at the following URL and choose a platform under the Cisco Access Products menu: https://mibs.cloudapps.cisco.com/ITDIT/MIBS/servlet/index |

RFCs

| RFCs | Title |
|----------|---|
| RFC 826 | Ethernet Address Resolution Protocol: Or converting network protocol addresses to 48.bit Ethernet address for transmission on Ethernet hardware |
| RFC 1027 | Using ARP to implement transparent subnet gateways |

Technical Assistance

| Description | Link |
|---|---|
| The Cisco Technical Support website contains thousands of pages of searchable technical content, including links to products, technologies, solutions, technical tips, and tools. Registered Cisco.com users can log in from this page to access even more content. | http://www.cisco.com/techsupport |



CHAPTER 4

Implementing the Dynamic Host Configuration Protocol

This module describes the concepts and tasks you will use to configure Dynamic Host Configuration Protocol (DHCP).

Feature History for Implementing the Dynamic Host Configuration Protocol

| Release | Modification |
|---------------|--|
| Release 2.0 | This feature was introduced. |
| Release 3.4.0 | The DHCP IPv6 Information Pool configuration procedure was added and DHCP relay information description was updated. |
| Release 3.7.0 | The DHCP CLI was modified. |

- [Prerequisites for Configuring DHCP Relay Agent](#), on page 43
- [Information About DHCP Relay Agent](#), on page 44
- [Secure ARP](#), on page 44
- [How to Configure and Enable DHCP Relay Agent](#), on page 44
- [DHCPv4 Client](#), on page 51
- [Information About Configuring DHCP IPv6 Information Pools](#), on page 52
- [How to Configure DHCP IPv6 Information Pools](#), on page 52
- [Enabling Secure ARP](#), on page 53
- [Configuration Examples for the DHCP Relay Agent](#), on page 54
- [Additional References](#), on page 56

Prerequisites for Configuring DHCP Relay Agent

The following prerequisites are required to configure a DHCP relay agent:

- You must be in a user group associated with a task group that includes the proper task IDs. The command reference guides include the task IDs required for each command. If you suspect user group assignment is preventing you from using a command, contact your AAA administrator for assistance.
- A configured and running DHCP client and DHCP server

- Connectivity between the relay agent and DHCP server

Information About DHCP Relay Agent

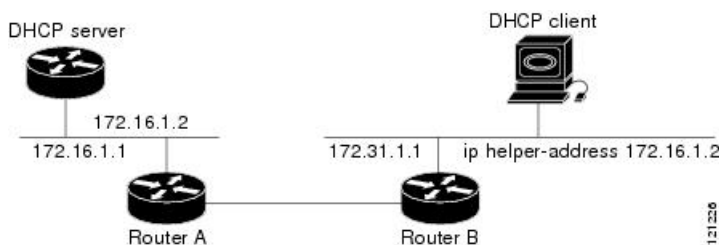
A DHCP relay agent is a host that forwards DHCP packets between clients and servers that do not reside on a shared physical subnet. Relay agent forwarding is distinct from the normal forwarding of an IP router where IP datagrams are switched between networks transparently.

DHCP clients use User Datagram Protocol (UDP) broadcasts to send DHCPDISCOVER messages when they lack information about the network to which they belong.

If a client is on a network segment that does not include a server, a relay agent is needed on that network segment to ensure that DHCP packets reach the servers on another network segment. UDP broadcast packets are not forwarded, because most routers are not configured to forward broadcast traffic. You can configure a DHCP relay profile and configure one or more helper addresses in it. You can assign the profile to an interface or a VRF.

[Figure 12: Forwarding UDP Broadcasts to a DHCP Server Using a Helper Address, on page 44](#) demonstrates the process. The DHCP client broadcasts a request for an IP address and additional configuration parameters on its local LAN. Acting as a DHCP relay agent, Router B picks up the broadcast, changes the destination address to the DHCP server's address and sends the message out on another interface. The relay agent inserts the IP address of the interface, on which the relay profile into the gateway address (giaddr) field of the DHCP packet, which enables the DHCP server to determine which subnet should receive the offer and identify the appropriate IP address range. The relay agent unicasts the messages to the server address, in this case 172.16.1.2 (which is specified by the helper address in the relay profile).

Figure 12: Forwarding UDP Broadcasts to a DHCP Server Using a Helper Address



Secure ARP

In standalone DHCP sessions, the DHCP server adds an ARP entry when it assigns an IP address to a client. However, in IP subscriber sessions, DHCP server does not add an ARP entry. Although ARP establishes correspondences between network addresses, an untrusted device can spoof IP an address not assigned to it posing a security threat for IP subscriber sessions. You can enable the secure ARP feature and allow DHCP to add an ARP cache entry when DHCP assigns an IP address to a client. Secure ARP is disabled by default.

How to Configure and Enable DHCP Relay Agent

This section contains the following tasks:

Configuring and Enabling DHCP Relay Agent with DHCP MAC Address Verification

This section discusses how to configure and enable DHCP Relay Agent with DHCP MAC address verification.

Configuration Example

```
Router# configure

Router(config)# dhcp ipv4
/* Configures DHCP for IPv4 and enters the DHCPv4 configuration submode. */

Router(config-dhcpv4)# profile client relay
/* Enables DHCP relay profile */

Router(config-dhcpv4)# client-mac-mismatch action drop
/* Enables MAC address verification. If MAC address in the DHCPv4 protocol header does not
match the L2 header source MAC address in the DHCPv4 relay profile,
the frame is dropped */

Router(config-dhcpv4-relay-profile)# relay information option
/* Inserts the DHCP relay agent information option (option-82 field) in forwarded
BOOTREQUEST messages to a DHCP server. */

Router(config-dhcpv4-relay-profile)# relay information check
/* (Optional) Configures DHCP to check the validity of the relay agent information
option in forwarded BOOTREPLY messages. */

Router(config-dhcpv4-relay-profile)# relay information policy drop
/* (Optional) Configures the reforwarding policy for a DHCP relay agent;
that is, whether the relay agent will drop or keep (using the 'keep' keyword)
the relay information. */

Router(config-dhcpv4-relay-profile)# relay information option allow-untrusted
/* (Optional) Configures the DHCP IPv4 Relay not to discard BOOTREQUEST packets that have
an existing
relay information option and the giaddr set to zero. */

Router(config-dhcpv4-relay-profile)# giaddr policy drop
/* Drops the packet that has an existing nonzero giaddr value. Use the 'replace' keyword
to replace the existing giaddr value with a value that it generates (the default behavior).
*/

Router(config-dhcpv4-relay-profile)# helper-address vrf vrf1 10.1.1.1
/* Forwards UDP broadcasts, including DHCP. */

Router(config-dhcpv4-relay-profile)# commit

Router(config-dhcpv4-relay-profile)# exit
Router(config-dhcpv4)# vrf vrf1 relay profile client
Router(config-dhcpv4)# commit
/* Configures DHCP Relay on a VRF and commits the entire configuration. */
```

Running Configuration

Confirm your configuration.

```
Router# show run
Thu May 11 09:00:57.839 IST
Building configuration...
```

```

!! IOS XR Configuration 0.0.0
!! Last configuration change at Thu May 11 09:00:54 2017 by annseque
!
dhcp ipv4
vrf vrf1 relay profile client
profile client relay
client-mac-match action drop
helper-address vrf vrf1 10.1.1.1
giaddr policy drop
relay information check
relay information option
relay information policy drop
relay information option allow-untrusted
!
!

```

DHCP MAC Address Verification

Use the following show command to check if DHCP MAC address is being verified on the router.

```

Router# show dhcp ipv4 relay statistics raw all
packet_drop_mac_mismatch           :           0

```

The output validates that the DHCP MAC address of the packets is verified.

Configuring the DHCPv6 (Stateless) Relay Agent

Perform this task to specify a destination address to which client messages are forwarded and to enable Dynamic Host Configuration Protocol (DHCP) for IPv6 relay service on the interface.

Configuration Example

To configure the DHCPv6 (stateless) relay agent, you must complete the following configurations:

1. Enable the DHCP IPv6 configuration mode.
2. Configure the DHCPv6 relay profile.
3. Configure helper addresses.
4. Specify the interface for the relay profile.

Configuration

```

/* Enter the global configuration mode, and then enter the DHCP IPv6 configuration mode */
Router# configure terminal
Router(config)# dhcp ipv6
Router(config-dhcpv6)# profile test relay
Router(config-dhcpv6-relay-profile)# helper-address vrf default 2001:1::1
Router(config-dhcpv6-relay-profile)# !
Router(config-dhcpv6-relay-profile)# interface TenGigE0/0/0/0 relay profile test
Router(config-dhcpv6)# !

```

Enabling DHCP Relay Agent on an Interface

This task describes how to enable the Cisco IOS XR DHCP relay agent on an interface.



Note On Cisco IOS XR software, the DHCP relay agent is disabled by default.

SUMMARY STEPS

1. `configure`
2. `dhcp ipv4`
3. `interface type name relay profile profile-name`
4. `commit`

DETAILED STEPS

| | Command or Action | Purpose |
|--------|--|---|
| Step 1 | <code>configure</code> | |
| Step 2 | <code>dhcp ipv4</code> Example: RP/0/RP0/CPU0:router(config-if)# <code>dhcp ipv4</code> | Enters DHCP IPv4 configuration submenu. |
| Step 3 | <code>interface type name relay profile profile-name</code> Example: RP/0/RP0/CPU0:router(config-dhcpv4)# <code>interface FastEthernet0/0 relay profile client</code> | Attaches a relay profile to an interface. |
| Step 4 | <code>commit</code> | |

Disabling DHCP Relay on an Interface

This task describes how to disable the DHCP relay on an interface by assigning the none profile to the interface.

SUMMARY STEPS

1. `configure`
2. `dhcp ipv4`
3. `interface type name none`
4. `commit`

DETAILED STEPS

| | Command or Action | Purpose |
|--------|------------------------|---------|
| Step 1 | <code>configure</code> | |

| | Command or Action | Purpose |
|---------------|--|---|
| Step 2 | dhcp ipv4 Example: RP/0/RP0/CPU0:router(config)# dhcp ipv4 | Enters DHCP IPv4 configuration submode. |
| Step 3 | interface type name none Example: RP/0/RP0/CPU0:router(config-dhcpv4-relay-profile)# interface pos 0/1/4/1 none | Disables the DHCP relay on the interface. |
| Step 4 | commit | |

Enabling DHCP Relay on a VRF

This task describes how to enable DHCP relay on a VRF.

SUMMARY STEPS

1. **configure**
2. **dhcp ipv4**
3. **vrf vrf-name relay profile profile-name**
4. **commit**

DETAILED STEPS

| | Command or Action | Purpose |
|---------------|---|---|
| Step 1 | configure | |
| Step 2 | dhcp ipv4 Example: RP/0/RP0/CPU0:router(config)# dhcp ipv4 | Enters DHCP IPv4 configuration submode. |
| Step 3 | vrf vrf-name relay profile profile-name Example: RP/0/RP0/CPU0:router(config-dhcpv4) #vrf default relay profile client | Enables DHCP relay on a VRF. |
| Step 4 | commit | |

Configuring the Relay Agent Information Feature

This task describes how to configure the DHCP relay agent information option processing capabilities.

A DHCP relay agent may receive a message from another DHCP relay agent that already contains relay information. By default, the relay information from the previous relay agent is replaced (using the replace option).

SUMMARY STEPS

1. **configure**
2. **dhcp ipv4**
3. **profile *profile-name* relay**
4. **relay information option**
5. **relay information check**
6. **relay information policy {drop | keep}**
7. **relay information option allow-untrusted**
8. **commit**

DETAILED STEPS

| | Command or Action | Purpose |
|--------|---|---|
| Step 1 | configure | |
| Step 2 | dhcp ipv4 Example: RP/0/RP0/CPU0:router(config)# dhcp ipv4 | Enters DHCP IPv4 configuration mode. |
| Step 3 | profile <i>profile-name</i> relay Example: RP/0/RP0/CPU0:router(config-dhcpv4)# profile client relay | Enters DHCP IPv4 profile relay mode. |
| Step 4 | relay information option Example: RP/0/RP0/CPU0:router(config-dhcpv4-relay-profile)# relay information option | Enables the system to insert the DHCP relay agent information option (option-82 field) in forwarded BOOTREQUEST messages to a DHCP server. <ul style="list-style-type: none"> • This option is injected by the relay agent while forwarding client-originated DHCP packets to the server. Servers recognizing this option can use the information to implement IP address or other parameter assignment policies. When replying, the DHCP server echoes the option back to the relay agent. The relay agent removes the option before forwarding the reply to the client. • The relay agent information is organized as a single DHCP option that contains one or more suboptions. These options contain the information known by the relay agent. The supported suboptions are: <ul style="list-style-type: none"> • Remote ID |

| | Command or Action | Purpose |
|---------------|---|--|
| | | <ul style="list-style-type: none"> • Circuit ID <p>Note This function is disabled by default.</p> |
| Step 5 | <p>relay information check</p> <p>Example:</p> <pre>RP/0/RP0/CPU0:router(config-dhcpv4-relay-profile)# relay information check</pre> | <p>(Optional) Configures DHCP to check that the relay agent information option in forwarded BOOTREPLY messages is valid.</p> <ul style="list-style-type: none"> • By default, DHCP checks that the option-82 field in DHCP reply packets, received from the DHCP server, is valid. If an invalid message is received, the relay agent drops the message. If a valid message is received, the relay agent removes the option-82 field and forwards the packet. <p>Note Use the relay information check command to reenale this functionality if the functionality has been disabled.</p> |
| Step 6 | <p>relay information policy {drop keep}</p> <p>Example:</p> <pre>RP/0/RP0/CPU0:router(config)# dhcp relay information policy drop</pre> | <p>(Optional) Configures the reforwarding policy for a DHCP relay agent; that is, whether the relay agent will drop or keep the relay information.</p> |
| Step 7 | <p>relay information option allow-untrusted</p> <p>Example:</p> <pre>RP/0/RP0/CPU0:router(config-dhcpv4-relay-profile)# relay information check</pre> | <p>(Optional) Configures the DHCP IPv4 Relay not to discard BOOTPREQUEST packets that have an existing relay information option and the giaddr set to zero.</p> |
| Step 8 | commit | |

Configuring Relay Agent Giaddr Policy

This task describes how to configure BOOTPREQUEST packets for Dynamic Host Configuration Protocol (DHCP) IPv4 Relay processes, that already contain a nonzero giaddr attribute.

SUMMARY STEPS

1. **configure**
2. **dhcp ipv4**
3. **profile *profile-name* relay**
4. **giaddr policy {replace | drop}**
5. **commit**

DETAILED STEPS

| | Command or Action | Purpose |
|--------|---|---|
| Step 1 | configure | |
| Step 2 | dhcp ipv4 Example: RP/0/RP0/CPU0:router(config)# dhcp ipv4 | Enables the DHCP IPv4 configuration submode. |
| Step 3 | profile profile-name relay Example: RP/0/RP0/CPU0:router(config-dhcpv4)# profile client relay | Enables profile relay submode. |
| Step 4 | giaddr policy {replace drop} Example: RP/0/RP0/CPU0:router(config-dhcpv4-relay-profile)# giaddr policy drop | Specifies the giaddr policy. <ul style="list-style-type: none"> • replace—Replaces the existing giaddr value with a value that it generates. • drop—Drops the packet that has an existing nonzero giaddr value. |
| Step 5 | commit | |

DHCPv4 Client

The Dynamic Host Configuration Protocol (DHCP) client functionality enables the router interfaces to dynamically acquire the IPv4 address using DHCP.

The DHCP provides configuration parameters to Internet hosts. DHCP consists of two components:

- a protocol to deliver host-specific configuration parameters from a DHCP server to a host.
- a mechanism to allocate 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.

A relay agent is required if the client and server are not on the same Layer 2 network. The relay agent usually runs on the router, and is required because the client device does not know its own IP address initially. The agent sends out a Layer 2 broadcast to find a server that has this information. The router relays these broadcasts to the DHCP server, and forwards the responses back to the correct Layer 2 address so that the correct device gets the correct configuration information.

DHCP has the ability to allocate IP addresses only for a configurable period of time, called the lease period. If the client is required to retain this IP address for a longer period beyond the lease period, the lease period must be renewed before the IP address expires. The client renews the lease based on configuration that was sent from the server. The client unicasts a REQUEST message using the IP address of the server. When a server receives the REQUEST message and responds with an ACK message. The lease period of the client is extended by the lease time configured in the ACK message.

Restrictions and Limitations

- DHCP client can be enabled only on management interfaces.
- Either DHCP or static IP can be configured on an interface.

Enabling DHCP Client on an Interface

The DHCPv4 or DHCPv6 client can be enabled at an interface level. The DHCP component receives a notification when DHCPv4 or DHCPv6 is enabled or disabled on an interface.

```
Router# configure
Router(config)# interface MgmtEth rack/slot/CPU0/port
Router(config)# interface interface_name ipv6 address dhcp
```

Information About Configuring DHCP IPv6 Information Pools

A *DHCP IPv6 configuration information pool* is a named entity that includes information about available configuration parameters and policies that control assignment of the parameters to clients from the pool. A pool is configured independently of the DHCP service and is associated with the DHCP service through the command line interface.

Each configuration pool can contain the following configuration parameters and operational information:

- Prefix delegation information, which could include a list of available prefixes for a particular client and associated preferred and valid lifetimes
- Domain name service (DNS) servers—List of IPv6 addresses of DNS servers
- Domain search list—String containing domain names for DNS resolution
- SIP server address—List of IPv6 addresses of SIP server
- SIP server domain list—String containing domain names for SIP server

How to Configure DHCP IPv6 Information Pools

This section contains the following task:

Configuring Cisco IOS XR DHCP IPv6 Information Pool Option

This task describes how to enable support for the DHCP IPv6 information pool option with the name pool1.

SUMMARY STEPS

1. **configure**
2. **dhcp ipv6**
3. **pool** *pool-name*
4. **commit**
5. **show dhcp ipv6 pool** [*pool-name*]

DETAILED STEPS

| | Command or Action | Purpose |
|--------|---|---|
| Step 1 | configure | |
| Step 2 | dhcp ipv6 Example: RP/0/RP0/CPU0:router (config)# dhcp ipv6 | Enables the DHCP IPv6 configuration mode. |
| Step 3 | pool <i>pool-name</i> Example: RP/0/RP0/CPU0:router (config-dhcp ipv6)# pool pool1 | Creates a DHCP pool specified by the <i>pool-name</i> argument for the prefix delegation and the other configurations on the interface. |
| Step 4 | commit | |
| Step 5 | show dhcp ipv6 pool [<i>pool-name</i>] Example: RP/0/RP0/CPU0:router# show dhcp ipv6 pool pool1 | (Optional) Displays the DHCP IPv6 pool name. |

Enabling Secure ARP

Secure ARP is disabled by default; this task describes how to enable secure ARP.

SUMMARY STEPS

1. **configure**
2. **dhcp ipv4**
3. Do one of the following:
 - **profile *profile-name* proxy**
 - **profile *profile-name* server**
4. **secure-arp**
5. **commit**

DETAILED STEPS

| | Command or Action | Purpose |
|--------|--|--------------------------------------|
| Step 1 | configure | |
| Step 2 | dhcp ipv4 Example: RP/0/RP0/CPU0:router(config)# dhcp ipv4 | Enters DHCP IPv4 configuration mode. |

| | Command or Action | Purpose |
|---------------|---|---|
| Step 3 | Do one of the following: <ul style="list-style-type: none"> • profile <i>profile-name</i> proxy • profile <i>profile-name</i> server Example: <pre>RP/0/RP0/CPU0:router(config-dhcpv4)# profile profile1 server</pre> | Enters DHCP IPv4 profile proxy or server submode. |
| Step 4 | secure-arp Example: <pre>RP/0/RP0/CPU0:router(config-dhcpv4-server-profile)# secure-arp</pre> | Enables secure ARP. |
| Step 5 | commit | |

Configuration Examples for the DHCP Relay Agent

This section provides the following configuration examples:

DHCP Relay Profile: Example

The following example shows how to configure the Cisco IOS XR relay profile:

```
dhcp ipv4
  profile client relay
    helper-address vrf foo 10.10.1.1
  !
  ! ...
```

DHCP Relay on an Interface: Example

The following example shows how to enable the DHCP relay agent on an interface:

```
dhcp ipv4
  interface GigabitEthernet 0/1/1/0 relay profile client
  !
```

DHCP Relay on a VRF: Example

The following example shows how to enable the DHCP relay agent on a VRF:

```
dhcp ipv4
  vrf default relay profile client
```

!

Relay Agent Information Option Support: Example

The following example shows how to enable the relay agent and the insertion and removal of the DHCP relay information option:

```
dhcp ipv4
  profile client relay
  relay information
  check
  !
  !
```

Relay Agent Giaddr Policy: Example

The following example shows how to configure relay agent giaddr policy:

```
dhcp ipv4
  profile client relay
  giaddr policy drop
  !
  !
```

Cisco IOS XR Broadcast Flag Policy: Example

This task describes how to configure DHCP IPv4 Relay to broadcast BOOTPREPLY packets only if the DHCP IPv4 broadcast flag is set in the DHCP IPv4 header.



Note By default, the DHCP IPv4 Relay always broadcasts BOOTPREPLY packets.

SUMMARY STEPS

1. **configure**
2. **dhcp ipv4**
3. **profile *profile name* relay**
4. **broadcast-flag policy check**
5. **commit**

DETAILED STEPS

| | Command or Action | Purpose |
|--------|-------------------|---------|
| Step 1 | configure | |

| | Command or Action | Purpose |
|--------|---|--|
| Step 2 | dhcp ipv4 Example: RP/0/RP0/CPU0:router(config)# dhcp ipv4 | Configures DHCP IPv4 mode. |
| Step 3 | profile profile name relay Example: RP/0/RP0/CPU0:router(config-dhcpv4)# profile client relay | Enables profile relay mode. |
| Step 4 | broadcast-flag policy check Example: RP/0/RP0/CPU0:router(config-dhcpv4-relay-profile)# broadcast-flag policy check | Enables checking of the broadcast flag in packets. |
| Step 5 | commit | |

Additional References

The following sections provide references related to implementing the Cisco IOS XR DHCP relay agent.

Related Documents

| Related Topic | Document Title |
|--|--|
| Cisco IOS XR DHCP commands | <i>DHCP Commands</i> module in the <i>IP Addresses and Services Command Reference for Cisco CRS Routers</i> |
| Getting started material | <i>Cisco IOS XR Getting Started Guide for the Cisco CRS Router</i> |
| Information about user groups and task IDs | <i>Configuring AAA Services</i> module in the <i>System Security Configuration Guide for Cisco CRS Routers</i> |

Standards

| Standards | Title |
|---|-------|
| No new or modified standards are supported by this feature, and support for existing standards has not been modified by this feature. | — |

MIBs

| MIBs | MIBs Link |
|-------------|---|
| — | To locate and download MIBs, use the Cisco MIB Locator found at the following URL and choose a platform under the Cisco Access Products menu: https://mibs.cloudapps.cisco.com/ITDIT/MIBS/servlet/index |

RFCs

| RFC | Title |
|-------------|--|
| RFC 2131 | <i>Dynamic Host Configuration Protocol</i> |
| RFC 3315 | <i>Dynamic Host Configuration Protocol for IPv6 (DHCPv6)</i> |

Technical Assistance

| Description | Link |
|---|---|
| The Cisco Technical Support website contains thousands of pages of searchable technical content, including links to products, technologies, solutions, technical tips, and tools. Registered Cisco.com users can log in from this page to access even more content. | http://www.cisco.com/techsupport |



CHAPTER 5

Implementing Host Services and Applications

Cisco IOS XR software Host Services and Applications features on the router are used primarily for checking network connectivity and the route a packet follows to reach a destination, mapping a hostname to an IP address or an IP address to a hostname, and transferring files between routers and UNIX workstations.



Note For detailed conceptual information about Cisco IOS XR software Host Services and Applications and complete descriptions of the commands listed in this module, see the

Host Services and Applications Commands module in *IP Addresses and Services Command Reference for Cisco CRS Routers*

Feature History for Implementing Host Services and Applications

| Release | Modification |
|-------------|------------------------------|
| Release 2.0 | This feature was introduced. |

- [Prerequisites for Implementing Host Services and Applications](#) , on page 59
- [Information About Implementing Host Services and Applications](#) , on page 60
- [How to Implement Host Services and Applications](#) , on page 63
- [Configuring syslog source-interface](#), on page 73
- [IPv6 Support for IP SLA ICMP Echo Operation](#), on page 74
- [Configuration Examples for Implementing Host Services and Applications](#) , on page 75
- [Additional References](#), on page 78

Prerequisites for Implementing Host Services and Applications

The following prerequisites are required to implement Cisco IOS XR software Host Services and applications

- You must be in a user group associated with a task group that includes the proper task IDs. The command reference guides include the task IDs required for each command. If you suspect user group assignment is preventing you from using a command, contact your AAA administrator for assistance.

Information About Implementing Host Services and Applications

To implement Cisco IOS XR software Host Services and applications features discussed in this document, you should understand the following concepts:

Key Features Supported in the Cisco IOS XR software Host Services and Applications Implementation

The following features are supported for host services and applications on Cisco IOS XR software:

- Ping and traceroute—The **ping** and **traceroute** commands are convenient, frequently used tools for checking network connectivity and troubleshooting network problems. The **ping** command determines whether a specific IP address is online by sending out a packet and waiting for a response. The **traceroute** command provides the path from the source to the remote destination being contacted.
- Domain services—The domain services act as a Berkeley Software Distribution (BSD) domain resolver. When an application requires the IP address of a hostname or the hostname of an IP address, the domain services attempt to find the address or hostname by checking the local cache. If there is no address entry in the cache, a Domain Name System (DNS) query is sent to the name server. After the address or hostname is retrieved from the name server, the address or hostname is given to the application.
- File transfer services (FTP, TFTP, and rcp)—FTP, TFTP, and rcp clients are implemented as resource managers. The resource managers are mainly used for transferring files to and from a remote host and to place core files on a remote host. See the *File System Commands* module of the *Cisco IOS XR System Management Configuration Guide for the Cisco CRS Router* for information on file transfer protocols.
- Cisco Inetd—Cisco Internet services daemon (Cinetd) is similar to UNIX inetd, in that it listens on a well-known port on behalf of the server program. When a service request is received on the port, Cinetd notifies the server program associated with the service request. By default, Cinetd is not configured to listen for any services. Cinetd is enabled by default. See the *Interface and Hardware Component Command Reference for Cisco CRS Routers* for information on supported Cinetd commands.

Network Connectivity Tools

Network connectivity tools enable you to check device connectivity by running traceroutes and pinging devices on the network.

Ping

The **ping** command is a common method for troubleshooting the accessibility of devices. It uses two Internet Control Message Protocol (ICMP) query messages, ICMP echo requests, and ICMP echo replies to determine whether a remote host is active. The **ping** command also measures the amount of time it takes to receive the echo reply.

The **ping** command first sends an echo request packet to an address, and then it waits for a reply. The ping is successful only if the echo request gets to the destination, and the destination is able to get an echo reply (hostname is alive) back to the source of the ping within a predefined time interval.

The bulk option has been introduced to check reachability to multiple destinations. The destinations are directly input through the CLI. This option is supported for ipv4 destinations only.

Traceroute

Where the **ping** command can be used to verify connectivity between devices, the **traceroute** command can be used to discover the paths packets take to a remote destination and where routing breaks down.

The **traceroute** command records the source of each ICMP "time-exceeded" message to provide a trace of the path that the packet took to reach the destination. You can use the IP **traceroute** command to identify the path that packets take through the network on a hop-by-hop basis. The command output displays all network layer (Layer 3) devices, such as routers, that the traffic passes through on the way to the destination.

The **traceroute** command uses the Time To Live (TTL) field in the IP header to cause routers and servers to generate specific return messages. The **traceroute** command sends a User Datagram Protocol (UDP) datagram to the destination host with the TTL field set to 1. If a router finds a TTL value of 1 or 0, it drops the datagram and sends back an ICMP time-exceeded message to the sender. The traceroute facility determines the address of the first hop by examining the source address field of the ICMP time-exceeded message.

To identify the next hop, the **traceroute** command sends a UDP packet with a TTL value of 2. The first router decrements the TTL field by 1 and sends the datagram to the next router. The second router sees a TTL value of 1, discards the datagram, and returns the time-exceeded message to the source. This process continues until the TTL increments to a value large enough for the datagram to reach the destination host (or until the maximum TTL is reached).

To determine when a datagram reaches its destination, the **traceroute** command sets the UDP destination port in the datagram to a very large value that the destination host is unlikely to be using. When a host receives a datagram with an unrecognized port number, it sends an ICMP port unreachable error to the source. This message indicates to the traceroute facility that it has reached the destination.

Domain Services

Cisco IOS XR software domain services acts as a Berkeley Standard Distribution (BSD) domain resolver. The domain services maintains a local cache of hostname-to-address mappings for use by applications, such as Telnet, and commands, such as **ping** and **traceroute**. The local cache speeds the conversion of hostnames to addresses. Two types of entries exist in the local cache: static and dynamic. Entries configured using the **domain ipv4 host** or **domain ipv6 host** command are added as static entries, while entries received from the name server are added as dynamic entries.

The name server is used by the World Wide Web (WWW) for translating names of network nodes into addresses. The name server maintains a distributed database that maps hostnames to IP addresses through the DNS protocol from a DNS server. One or more name servers can be specified using the **domain name-server** command.

When an application needs the IP address of a host or the hostname of an IP address, a remote-procedure call (RPC) is made to the domain services. The domain service looks up the IP address or hostname in the cache, and if the entry is not found, the domain service sends a DNS query to the name server.

You can specify a default domain name that Cisco IOS XR software uses to complete domain name requests. You can also specify either a single domain or a list of domain names. Any IP hostname that does not contain a domain name has the domain name you specify appended to it before being added to the host table. To specify a domain name or names, use either the **domain name** or **domain list** command.

TFTP Server

It is too costly and inefficient to have a machine that acts only as a server on every network segment. However, when you do not have a server on every segment, your network operations can incur substantial time delays

across network segments. You can configure a router to serve as a TFTP server to reduce costs and time delays in your network while allowing you to use your router for its regular functions.

Typically, a router that is configured as a TFTP server provides other routers with system image or router configuration files from its flash memory. You can also configure the router to respond to other types of services requests.

File Transfer Services

File Transfer Protocol (FTP), Trivial File Transfer Protocol (TFTP), remote copy protocol (rcp) rcp clients, and Secure Copy Protocol (SCP) are implemented as file systems or resource managers. For example, pathnames beginning with `tftp://` are handled by the TFTP resource manager.

The file system interface uses URLs to specify the location of a file. URLs commonly specify files or locations on the WWW. However, on Cisco routers, URLs also specify the location of files on the router or remote file servers.

When a router crashes, it can be useful to obtain a copy of the entire memory contents of the router (called a core dump) for your technical support representative to use to identify the cause of the crash. SCP, FTP, TFTP, or rcp can be used to save the core dump to a remote server. See the *System Management Configuration Guide for Cisco CRS Routers* for information on executing a core dump.

RCP

The remote copy protocol (RCP) commands rely on the remote shell (rsh) server (or daemon) on the remote system. To copy files using rcp, you do not need to create a server for file distribution, as you do with TFTP. You need only to have access to a server that supports the rsh. Because you are copying a file from one place to another, you must have read permissions for the source file and write permission in the destination directory. If the destination file does not exist, rcp creates it for you.

Although Cisco rcp implementation emulates the functions of the UNIX rcp implementation—copying files among systems on the network—Cisco command syntax differs from the UNIX rcp command syntax. Cisco IOS XR software offers a set of copy commands that use rcp as the transport mechanism. These **rcp copy** commands are similar in style to the Cisco IOS XR software TFTP copy commands, but they offer an alternative that provides faster performance and reliable delivery of data. These improvements are possible because the rcp transport mechanism is built on and uses the TCP/IP stack, which is connection-oriented. You can use rcp commands to copy system images and configuration files from the router to a network server and so forth.

FTP

File Transfer Protocol (FTP) is part of the TCP/IP protocol stack, which is used for transferring files between network nodes. FTP is defined in RFC 959.

TFTP

Trivial File Transfer Protocol (TFTP) is a simplified version of FTP that allows files to be transferred from one computer to another over a network, usually without the use of client authentication (for example, username and password).

SCP

Secure Copy Protocol (SCP) is a file transfer protocol which provides a secure and authenticated method for transferring files. SCP relies on SSHv2 to transfer files from a remote location to a local location or from local location to a remote location.

Cisco IOS XR software supports SCP server and client operations. If a device receives an SCP request, the SSH server process spawns the SCP server process which interacts with the client. For each incoming SCP subsystem request, a new SCP server instance is spawned. If a device sends a file transfer request to a destination device, it acts as the client.

When a device starts an SSH connection to a remote host for file transfer, the remote device can either respond to the request in Source Mode or Sink Mode. In Source Mode, the device is the file source. It reads the file from its local directory and transfers the file to the intended destination. In Sink Mode, the device is the destination for the file to be transferred.

Using SCP, you can copy a file from the local device to a destination device or from a destination device to the local device.

Using SCP, you can only transfer individual files. You cannot transfer a file from a destination device to another destination device.

Cisco inetd

Cisco Internet services process daemon (Cinetd) is a multithreaded server process that is started by the system manager after the system has booted. Cinetd listens for Internet services such as Telnet service, TFTP service, and so on. Whether Cinetd listens for a specific service depends on the router configuration. For example, when the **tftp server** command is entered, Cinetd starts listening for the TFTP service. When a request arrives, Cinetd runs the server program associated with the service.

Telnet

Enabling Telnet allows inbound Telnet connections into a networking device.

How to Implement Host Services and Applications

This section contains the following procedures:

Checking Network Connectivity

As an aid to diagnosing basic network connectivity, many network protocols support an echo protocol. The protocol involves sending a special datagram to the destination host, then waiting for a reply datagram from that host. Results from this echo protocol can help in evaluating the path-to-host reliability, delays over the path, and whether the host can be reached or is functioning.

SUMMARY STEPS

1. **ping** [**ipv4** | **ipv6** | **vrf** *vrf-name*] [*host-name* | *ip-address*]

DETAILED STEPS

| | Command or Action | Purpose |
|---------------|--|--|
| Step 1 | <p>ping [ipv4 ipv6 vrf <i>vrf-name</i>] [<i>host-name</i> <i>ip-address</i>]</p> <p>Example:</p> <pre>RP/0/RP0/CPU0:router# ping</pre> | <p>Starts the ping tool that is used for testing connectivity.</p> <p>Note If you do not enter a hostname or an IP address on the same line as the ping command, the system prompts you to specify the target IP address and several other command parameters. After specifying the target IP address, you can specify alternate values for the remaining parameters or accept the displayed default for each parameter.</p> |

Checking Network Connectivity for Multiple Destinations

The bulk option enables you to check reachability to multiple destinations. The destinations are directly input through the CLI. This option is supported for ipv4 destinations only.

SUMMARY STEPS

1. **ping bulk ipv4** [**input cli** { **batch** | **inline** }]
2. [**vrf** *vrf-name*] [*host-name* | *ip-address*]

DETAILED STEPS

| | Command or Action | Purpose |
|---------------|---|--|
| Step 1 | <p>ping bulk ipv4 [input cli { batch inline }]</p> <p>Example:</p> <pre>RP/0/RP0/CPU0:router# ping bulk ipv4 input cli</pre> | Starts the ping tool that is used for testing connectivity. |
| Step 2 | <p>[vrf <i>vrf-name</i>] [<i>host-name</i> <i>ip-address</i>]</p> <p>Example:</p> <pre>Please enter input via CLI with one destination per line: vrf myvrf1 1.1.1.1 vrf myvrf2 2.2.2.2 vrf myvrf1 myvrf1.cisco.com vrf myvrf2 myvrf2.cisco.com Starting pings... Type escape sequence to abort. Sending 1, 100-byte ICMP Echos to 1.1.1.1, vrf is myvrf1: ! Success rate is 100 percent (1/1), round-trip min/avg/max = 1/1/1 ms</pre> | You must hit the Enter button and then specify one destination address per line. |

| | Command or Action | Purpose |
|--|---|---------|
| | <pre> Sending 2, 100-byte ICMP Echos to 2.2.2.2, vrf is myvrf2: !! Success rate is 100 percent (2/2), round-trip min/avg/max = 1/1/1 ms Sending 1, 100-byte ICMP Echos to 1.1.1.1, vrf is myvrf1: ! Success rate is 100 percent (1/1), round-trip min/avg/max = 1/4/1 ms Sending 2, 100-byte ICMP Echos to 2.2.2.2, vrf is myvrf2: !! Success rate is 100 percent (2/2), round-trip min/avg/max = 1/3/1 ms </pre> | |

Checking Packet Routes

The **tracert** command allows you to trace the routes that packets actually take when traveling to their destinations.

SUMMARY STEPS

1. **tracert** [ipv4 | ipv6 | vrf vrf-name] [host-name | ip-address]

DETAILED STEPS

| | Command or Action | Purpose |
|---------------|---|---|
| Step 1 | <p>tracert [ipv4 ipv6 vrf vrf-name] [host-name ip-address]</p> <p>Example:</p> <pre>RP/0/RP0/CPU0:router# tracert</pre> | <p>Traces packet routes through the network.</p> <p>Note If you do not enter a hostname or an IP address on the same line as the tracert command, the system prompts you to specify the target IP address and several other command parameters. After specifying the target IP address, you can specify alternate values for the remaining parameters or accept the displayed default for each parameter.</p> |

Configuring Domain Services

This task allows you to configure domain services.

Before you begin

DNS-based hostname-to-address translation is enabled by default. If hostname-to-address translation has been disabled using the **domain lookup disable** command, re-enable the translation using the **no domain lookup disable** command. See the *IP Addresses and Services Command Reference for Cisco CRS Routers* for more information on the **domain lookup disable** command.

SUMMARY STEPS

1. **configure**
2. Do one of the following:
 - **domain name** *domain-name*
 - or
 - **domain list** *domain-name*
3. **domain name-server** *server-address*
4. **domain** {*ipv4* | *ipv6*} **host** *host-name* {*ipv4address* | *ipv6address*}
5. **commit**

DETAILED STEPS

| | Command or Action | Purpose |
|---------------|--|--|
| Step 1 | configure | |
| Step 2 | Do one of the following: <ul style="list-style-type: none"> • domain name <i>domain-name</i> • or • domain list <i>domain-name</i> Example: <pre>RP/0/RP0/CPU0:router(config)# domain name cisco.com or RP/0/RP0/CPU0:router(config)# domain list domain1.com</pre> | Defines a default domain name used to complete unqualified hostnames. |
| Step 3 | domain name-server <i>server-address</i> Example: <pre>RP/0/RP0/CPU0:router(config)# domain name-server 192.168.1.111</pre> | Specifies the address of a name server to use for name and address resolution (hosts that supply name information). Note You can enter up to six addresses, but only one for each command. |
| Step 4 | domain { <i>ipv4</i> <i>ipv6</i> } host <i>host-name</i> { <i>ipv4address</i> <i>ipv6address</i> } Example: <pre>RP/0/RP0/CPU0:router(config)# domain ipv4 host1 192.168.7.18</pre> | (Optional) Defines a static hostname-to-address mapping in the host cache using IPv4. Note You can bind up to eight additional associated addresses to a hostname. |
| Step 5 | commit | |

Configuring a Router as a TFTP Server

This task allows you to configure the router as a TFTP server so other devices acting as TFTP clients are able to read and write files from and to the router under a specific directory, such as slot0:/tmp, and so on (TFTP home directory).



Note For security reasons, the TFTP server requires that a file must already exist for a write request to succeed.

Before you begin

The server and client router must be able to reach each other before the TFTP function can be implemented. Verify this connection by testing the connection between the server and client router (in either direction) using the `ping` command.

SUMMARY STEPS

1. `configure`
2. `tftp {ipv4 | ipv6} server {homedir tftp-home-directory} {max-servers number} [access-list name]`
3. `commit`
4. `show cinetd services`

DETAILED STEPS

| | Command or Action | Purpose |
|--------|--|--|
| Step 1 | <code>configure</code> | |
| Step 2 | <p><code>tftp {ipv4 ipv6} server {homedir tftp-home-directory} {max-servers number} [access-list name]</code></p> <p>Example:</p> <pre>RP/0/RP0/CPU0:router(config)# tftp ipv4 server access-list listA homedir disk0</pre> | <p>Specifies:</p> <ul style="list-style-type: none"> • IPv4 or IPv6 address prefixes (required) • Home directory (required) • Maximum number of concurrent TFTP servers (required) • Name of the associated access list (optional) |
| Step 3 | <code>commit</code> | |
| Step 4 | <p><code>show cinetd services</code></p> <p>Example:</p> <pre>RP/0/RP0/CPU0:router# show cinetd services</pre> | Displays the network service for each process. The service column shows TFTP if the TFTP server is configured. |

Configuring a Router to Use rcp Connections

This task allows you to configure a router to use rcp.

Before you begin

For the rcp copy request to execute successfully, an account must be defined on the network server for the remote username.

If you are reading or writing to the server, the rcp server must be properly configured to accept the rcp read/write request from the user on the router. For UNIX systems, you must add an entry to the hosts file for the remote user on the rcp server.

SUMMARY STEPS

1. **configure**
2. **rcp client username** *username*
3. **rcp client source-interface** *type interface-path-id*
4. **commit**

DETAILED STEPS

| | Command or Action | Purpose |
|---------------|---|--|
| Step 1 | configure | |
| Step 2 | rcp client username <i>username</i> Example: RP/0/RP0/CPU0:router(config)# rcp client username netadmin1 | Specifies the name of the remote user on the rcp server. This name is used when a remote copy using rcp is requested. If the rcp server has a directory structure, all files and images to be copied are searched for or written relative to the directory in the remote user account. |
| Step 3 | rcp client source-interface <i>type interface-path-id</i> Example: RP/0/RP0/CPU0:router(config)# rcp client source-interface gigabitethernet 1/0/2/1 | Sets the IP address of an interface as the source for all rcp connections. |
| Step 4 | commit | |

Troubleshooting Tips

When using rcp to copy any file from a source to a destination, use the following path format:

```
copy rcp
:
//username
@
{
hostname
|
ipaddress
}/
directory-path
/
pie-name target-device
```

When using an IPv6 rcp server, use the following path format:

```
copy rcp
```

```

:
//username
@
[ipv6-address]/
directory-path
/
pie-name

```

See the **copy** command in the *System Management Command Reference for Cisco CRS Routers* for detailed information on using rcp protocol with the **copy** command.

Configuring a Router to Use FTP Connections

This task allows you to configure the router to use FTP connections for transferring files between systems on the network. With the the Cisco IOS XR Softwareimplementation of FTP, you can set the following FTP characteristics:

- Passive-mode FTP
- Password
- IP address

SUMMARY STEPS

1. **configure**
2. **ftp client passive**
3. **ftp client anonymous-password** *password*
4. **ftp client source-interface** *type interface-path-id*
5. **commit**

DETAILED STEPS

| | Command or Action | Purpose |
|--------|---|--|
| Step 1 | configure | |
| Step 2 | ftp client passive Example: RP/0/RP0/CPU0:router(config)# ftp client passive | Allows the software to use only passive FTP connections. |
| Step 3 | ftp client anonymous-password <i>password</i> Example: RP/0/RP0/CPU0:router(config)# ftp client anonymous-password xxxx | Specifies the password for anonymous users. |
| Step 4 | ftp client source-interface <i>type interface-path-id</i> Example: | Specifies the source IP address for FTP connections. |

| | Command or Action | Purpose |
|---------------|--|---------|
| | RP/0/RP0/CPU0:router(config)# ftp client source-interface GigabitEthernet 0/1/2/1 | |
| Step 5 | commit | |

Troubleshooting Tips

When using FTP to copy any file from a source to a destination, use the following path format:

```
copy ftp
://
username:password
@
{
hostname
|
ipaddress
}/
directory-path
/
pie-name target-device
```

When using an IPv6 FTP server, use the following path format:

```
copy ftp
:
//username
:
password
@
[ipv6-address]/
directory-path
/
pie-name
```

If unsafe or reserved characters appear in the username, password, hostname, and so on, they have to be encoded (RFC 1738).

The following characters are unsafe:

```
"<", ">", "#", "%", "\", "}", "|", "□", "~", "[", "]", and "\"
```

The following characters are reserved:

```
":", "/" "?", ":", "@", and "&"
```

The *directory-path* is a relative path to the home directory of the user. The slash (/) has to be encoded as %2f to specify the absolute path. For example:

```
ftp://user:password@hostname/%2fTFTPboot/directory/pie-name
```

See the **copy** command in the *System Management Command Reference for Cisco CRS Routers* for detailed information on using FTP protocol with the **copy** command.

Configuring a Router to Use TFTP Connections

This task allows you to configure a router to use TFTP connections. You must specify the source IP address for a TFTP connection.

SUMMARY STEPS

1. **configure**
2. **tftp client source-interface** *type*
3. **commit**

DETAILED STEPS

| | Command or Action | Purpose |
|--------|---|---|
| Step 1 | configure | |
| Step 2 | tftp client source-interface <i>type</i> Example: RP/0/RP0/CPU0:router(config)# tftp client source-interface GigabitEthernet 1/0/2/1 | Specifies the source IP address for TFTP connections. |
| Step 3 | commit | |

Troubleshooting Tips

When using TFTP to copy any file from a source to a destination, use the following path format:

```
copy tftp
://{
hostname
|
ipaddress
}/
directory-path
/
pie-name target-device
```

When using an IPv6 TFTP server, use the following path format:

```
copy tftp
:
//
[ipv6-address]/
directory-path
```

```

/
pie-name

```

See the **copy** command in the *System Management Command Reference for Cisco CRS Routers* for detailed information on using TFTP protocol with the **copy** command.

Configuring Telnet Services

This task allows you to configure Telnet services.

SUMMARY STEPS

1. **configure**
2. **telnet [ipv4 | ipv6 | vrf vrf-name] server max-servers 1**
3. **commit**

DETAILED STEPS

| | Command or Action | Purpose |
|---------------|--|--|
| Step 1 | configure | |
| Step 2 | telnet [ipv4 ipv6 vrf vrf-name] server max-servers 1 Example: RP/0/RP0/CPU0:router(config)# telnet ipv4 server max-servers 1 | Enables one inbound Telnet IPv4 server on the router. Note This command affects only inbound Telnet connections to the router. |
| Step 3 | commit | |

Transferring Files Using SCP

Secure Copy Protocol (SCP) allows you to transfer files between source and destination devices.

SUMMARY STEPS

1. Do one of the following:
 - **scp local-directory/filename username@location/directory/filename**
 - **scp username@location/directory/filename local-directory/filename**

DETAILED STEPS

| | Command or Action | Purpose |
|---------------|---|--|
| Step 1 | Do one of the following: <ul style="list-style-type: none"> • scp local-directory/filename username@location/directory/filename • scp username@location/directory/filename local-directory/filename | Use the scp local-directory/filename username@location/directory/filename command to transfer a file from a local directory to a remote directory. Use the scp username@location/directory/filename local-directory/filename to transfer a file from a remote directory to a local directory. |

| | Command or Action | Purpose |
|--|--|--|
| | Example: RP/0/RP0/CPU0:router# scp /usr/file1.txt root@209.165.200.1:/root/file3.txt or RP/0/RP0/CPU0:router# scp root@209.165.200.1:/root/file4.txt /usr/file.txt | You can transfer one file at a time. If the destination is a server, SSH server process must be running. |

Configuring syslog source-interface

Perform this task to configure the logging source interface to identify the syslog traffic, originating in a VRF from a particular router, as coming from a single device.

SUMMARY STEPS

1. **configure**
2. **logging source-interface** *interface vrf vrf-name*
3. **commit**
4. **show running-configuration logging**

DETAILED STEPS

| | Command or Action | Purpose |
|---------------|---|--|
| Step 1 | configure | |
| Step 2 | logging source-interface <i>interface vrf vrf-name</i> Example: RP/0/RP0/CPU0:router(config)# logging source-interface loopback 0 vrf vrf1 RP/0/RP0/CPU0:router(config)# logging source-interface loopback 1 vrf default | Configures the logging source interface to identify the syslog traffic, originating in a VRF from a particular router, as coming from a single device. |
| Step 3 | commit | |
| Step 4 | show running-configuration logging Example: RP/0/RP0/CPU0:router(config)# exit RP/0/RP0/CPU0:router# show running-configuration logging logging trap debugging logging 223.255.254.249 vrf vrf1 logging 223.255.254.248 vrf default logging source-interface Loopback0 vrf vrf1 logging source-interface Loopback1 | Verifies that the logging source is correctly configured for the VRF. |

IPv6 Support for IP SLA ICMP Echo Operation

IP Service Level Agreements (SLAs) Internet Control Message Protocol (ICMP) Echo operation is used to monitor the end-to-end response time between a Cisco router and devices using IP. ICMP Echo is useful for troubleshooting network connectivity issues.

Configuring an IPSLA ICMP echo operation

To monitor IP connections on a device, use the IP SLA ICMP Echo operation. This operation does not require the IP SLAs Responder to be enabled.

SUMMARY STEPS

1. **configure**
2. **ipsla**
3. **operation *n***
4. **type icmp echo**
5. **timeout *n***
6. **source address *address***
7. **destination address *address***
8. **commit**

DETAILED STEPS

| | Command or Action | Purpose |
|--------|--|---|
| Step 1 | configure | |
| Step 2 | ipsla Example: <pre>RP/0/RP0/CPU0:router(config)# ipsla</pre> | Enters IP SLA monitor configuration mode. |
| Step 3 | operation <i>n</i> Example: <pre>RP/0/RP0/CPU0:router(config-ipsla)# operation 500</pre> | Initiates configuration for an IP SLA operation. |
| Step 4 | type icmp echo Example: <pre>RP/0/RP0/CPU0:router(config-ipsla-op)# type icmp echo</pre> | Enters IP SLA ICMP Echo configuration mode. |
| Step 5 | timeout <i>n</i> Example: | Sets the timeout in ms. The default is 5000 milliseconds. |

| | Command or Action | Purpose |
|---------------|---|---|
| | <pre>RP/0/RP0/CPU0:router(config-ipsla-icmp-echo)# timeout 1000</pre> | |
| Step 6 | <p>source address <i>address</i></p> <p>Example:</p> <pre>RP/0/RP0/CPU0:router(config-ipsla-icmp-echo)# source address fe80::226:98ff:fe2e:3287</pre> | Configures the address of the source device. |
| Step 7 | <p>destination address <i>address</i></p> <p>Example:</p> <pre>RP/0/RP0/CPU0:router(config-ipsla-icmp-echo)# destination address fe80::226:98ff:fe2e:3287</pre> | Configures the address of the destination device. |
| Step 8 | commit | |

Configuration Examples for Implementing Host Services and Applications

This section provides the following configuration examples:

Checking Network Connectivity: Example

The following example shows an extended **ping** command sourced from the Router A Ethernet 0 interface and destined for the Router B Ethernet interface. If this ping succeeds, it is an indication that there is no routing problem. Router A knows how to get to the Ethernet of Router B, and Router B knows how to get to the Ethernet of Router A. Also, both hosts have their default gateways set correctly.

If the extended **ping** command from Router A fails, it means that there is a routing problem. There could be a routing problem on any of the three routers: Router A could be missing a route to the subnet of Router B's Ethernet, or to the subnet between Router C and Router B; Router B could be missing a route to the subnet of Router A's subnet, or to the subnet between Router C and Router A; and Router C could be missing a route to the subnet of Router A's or Router B's Ethernet segments. You should correct any routing problems, and then Host 1 should try to ping Host 2. If Host 1 still cannot ping Host 2, then both hosts' default gateways should be checked. The connectivity between the Ethernet of Router A and the Ethernet of Router B is checked with the extended **ping** command.

With a normal ping from Router A to Router B's Ethernet interface, the source address of the ping packet would be the address of the outgoing interface; that is, the address of the serial 0 interface (172.31.20.1). When Router B replies to the ping packet, it replies to the source address (that is, 172.31.20.1). This way, only the connectivity between the serial 0 interface of Router A (172.31.20.1) and the Ethernet interface of Router B (192.168.40.1) is tested.

To test the connectivity between Router A's Ethernet 0 (172.16.23.2) and Router B's Ethernet 0 (192.168.40.1), we use the extended **ping** command. With extended **ping**, we get the option to specify the source address of the **ping** packet.

In this example, the extended **ping** command verifies the IP connectivity between the two IP addresses 10.0.0.2 and 10.0.0.1.

```
ping

Protocol [ip]:
Target IP address: 10.0.0.1
Repeat count [5]:
Datagram size [100]:
Timeout in seconds [2]:
Extended commands? [no]: yes
Source address or interface: 10.0.0.2
Type of service [0]:
Set DF bit in IP header? [no]:
Validate reply data? [no]: yes
Data pattern [0xABCD]:
Loose, Strict, Record, Timestamp, Verbose[none]:
Sweep range of sizes? [no]:
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 10.25.58.21, timeout is 2 seconds:
!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 2/11/49 ms
```

The **traceroute** command is used to discover the paths packets take to a remote destination and where routing breaks down. The **traceroute** command provides the path between the two IP addresses and does not indicate any problems along the path.

```
traceroute

Protocol [ip]:
Target IP address: ena-view3
Source address: 10.0.58.29
Numeric display? [no]:
Timeout in seconds [3]:
Probe count [3]:
Minimum Time to Live [1]:
Maximum Time to Live [30]:
Port Number [33434]:
Loose, Strict, Record, Timestamp, Verbose[none]:

Type escape sequence to abort.
Tracing the route to 171.71.164.199

 1 sjc-jpollock-vpn.cisco.com (10.25.0.1) 30 msec 4 msec 4 msec
 2 15lab-vlan525-gw1.cisco.com (172.19.72.2) 7 msec 5 msec 5 msec
 3 sjc15-00lab-gw1.cisco.com (172.24.114.33) 5 msec 6 msec 6 msec
 4 sjc5-lab4-gw1.cisco.com (172.24.114.89) 5 msec 5 msec 5 msec
 5 sjc5-sbb4-gw1.cisco.com (171.71.241.162) 5 msec 6 msec 6 msec
 6 sjc5-dc5-gw1.cisco.com (171.71.241.10) 6 msec 6 msec 5 msec
 7 sjc5-dc1-gw1.cisco.com (171.71.243.2) 7 msec 8 msec 8 msec
 8 ena-view3.cisco.com (171.71.164.199) 6 msec * 8 msec
```

Configuring Domain Services: Example

The following example shows how to configure domain services on a router.

Defining the Domain Host

```
configure
domain ipv4 host host1 192.168.7.18
domain ipv4 host host2 10.2.0.2 192.168.7.33
```

Defining the Domain Name

```
configure
domain name cisco.com
```

Specifying the Addresses of the Name Servers

```
configure
domain name-server 192.168.1.111
domain name-server 192.168.1.2
```

Configuring a Router to Use rcp, FTP, or TFTP Connections: Example

The following example shows how to configure the router to use rcp, FTP, or TFTP connections.

Using rcp

```
configure
rcp client username netadmin1
rcp client source-interface gigabitethernet 1/0/2/1
```

Using FTP

```
configure
ftp client passive
ftp client anonymous-password xxxx
ftp client source-interface gigabitethernet 0/1/2/1
```

Using TFTP

```
configure
tftp client source-interface gigabitethernet 1/0/2/1
```

Additional References

The following sections provide references related to implementing host services and addresses on the Cisco IOS XR Software.

Related Documents

| Related Topic | Document Title |
|---|---|
| Host services and applications commands | <i>Host Services and Applications Commands</i> module in <i>IP Addresses and Services Command Reference for Cisco CRS Routers</i> |

Standards

| Standards | Title |
|---|-------|
| No new or modified standards are supported by this feature, and support for existing standards has not been modified by this feature. | — |

MIBs

| MIBs | MIBs Link |
|------|--|
| — | To locate and download MIBs using Cisco IOS XR software, use the Cisco MIB Locator found at the following URL and choose a platform under the Cisco Access Products menu: https://mibs.cloudapps.cisco.com/ITDIT/MIBS/servlet/index |

RFCs

| RFCs | Title |
|-----------------------|---------------------------------|
| RFC-959 | File Transfer Protocol |
| RFC-1738 and RFC-2732 | Uniform Resource Locators (URL) |
| RFC-783 | Trivial File Transfer Protocol |

Technical Assistance

| Description | Link |
|---|---|
| The Cisco Technical Support website contains thousands of pages of searchable technical content, including links to products, technologies, solutions, technical tips, and tools. Registered Cisco.com users can log in from this page to access even more content. | http://www.cisco.com/techsupport |



CHAPTER 6

Prerequisites for Implementing Access Lists and Prefix Lists

The following prerequisite applies to implementing access lists and prefix lists:

All command task IDs are listed in individual command references and in the Cisco IOS XR Task ID Reference Guide. If you need assistance with your task group assignment, contact your system administrator.

- [Restrictions for Implementing Access Lists and Prefix Lists, on page 79](#)
- [Restrictions for Implementing ACL-based Forwarding, on page 80](#)
- [Information About Implementing Access Lists and Prefix Lists , on page 80](#)
- [Information About Implementing ACL-based Forwarding, on page 89](#)
- [How to Implement Access Lists and Prefix Lists , on page 89](#)
- [How to Implement ACL-based Forwarding, on page 102](#)
- [Configuring Pure ACL-Based Forwarding for IPv6 ACL, on page 104](#)
- [Configuring Pure ACL-based Forwarding for ACL, on page 105](#)
- [ACL-Chaining, on page 106](#)
- [Configuration Examples for Implementing Access Lists and Prefix Lists , on page 108](#)
- [Configuration Examples for Implementing ACL-based Forwarding, on page 109](#)
- [IPv6 ACL in Class Map, on page 111](#)
- [Configuring an Interface to accept Common ACL - Examples, on page 114](#)
- [Additional References, on page 116](#)

Restrictions for Implementing Access Lists and Prefix Lists

The following restrictions apply to implementing access lists and prefix lists:

- Layer 2/Layer 3 ACLs are not supported on Layer 2 interfaces.
- Object group ACLs are not supported.
- IPv4 ACLs are not supported for loopback and interflex interfaces.
- IPv6 ACLs are not supported for loopback, interflex and L2 Ethernet Flow Point (EFP) main or subinterfaces.
- IPv6 ACL configuration on bundle interfaces (Ethernet LAG bundles only) is not supported.

- If the TCAM utilization is high and large ACLs are modified, then an error may occur. During such instances, do the following to edit an ACL:



-
- Note**
1. Remove the ACL from the interface.
 2. Reconfigure the ACL.
 3. Reapply the ACL to the interface.

Use the **show prm server team summary all acl all location** and **show pfilter-ea fea summary location** commands to view the TCAM utilization.

- Filtering of MPLS packets through common ACL and interface ACL is not supported.
- Video Monitoring is not supported through ACLs on IPv6 interfaces.
- You can configure an ACL name with a maximum of 64 characters.
- You can configure an ACL name to comprise of only letters and numbers.

Restrictions for Implementing ACL-based Forwarding

The following restrictions apply to implementing ACL-based forwarding (ABF):

- No support for IPv4 multicast traffic.
- No support for ACL-based forwarding from a software switching path (for example, IPv4 option packets).
- Support is only on physical interfaces, subinterfaces, and bundles.
- ACL-based forwarding is an ingress-only feature.
- When optional parameter interface-statistics is provided, the system pre-allocates a fixed number of contiguous statistic entries per ACE. The maximum number of contiguous statistic entries with unique counters for each ACL, of each direction of each line card, can be up to eight entries.

Information About Implementing Access Lists and Prefix Lists

To implement access lists and prefix lists, you must understand the following concepts:

Access Lists and Prefix Lists Feature Highlights

This section lists the feature highlights for access lists and prefix lists.

- Cisco IOS XR software provides the ability to clear counters for an access list or prefix list using a specific sequence number.
- Cisco IOS XR software provides the ability to copy the contents of an existing access list or prefix list to another access list or prefix list.

- Cisco IOS XR software allows users to apply sequence numbers to permit or deny statements and to resequence, add, or remove such statements from a named access list or prefix list.



Note Resequencing is only for IPv4 prefix lists.

- Cisco IOS XR software does not differentiate between standard and extended access lists. Standard access list support is provided for backward compatibility.
- Cisco IOS XR software provides ACL-based forwarding to forward packets to a next-hop that is specified by the ACL rule.
- Atomic update is supported for both ACL and ACL-based forwarding.

Purpose of IP Access Lists

Access lists perform packet filtering to control which packets move through the network and where. Such controls help to limit network traffic and restrict the access of users and devices to the network. Access lists have many uses, and therefore many commands accept a reference to an access list in their command syntax. Access lists can be used to do the following:

- Filter incoming packets on an interface.
- Filter outgoing packets on an interface.
- Restrict the contents of routing updates.
- Limit debug output based on an address or protocol.
- Control vty access.
- Identify or classify traffic for advanced features, such as congestion avoidance, congestion management, and priority and custom queueing.

How an IP Access List Works

An access list is a sequential list consisting of permit and deny statements that apply to IP addresses and possibly upper-layer IP protocols. The access list has a name by which it is referenced. Many software commands accept an access list as part of their syntax.

An access list can be configured and named, but it is not in effect until the access list is referenced by a command that accepts an access list. Multiple commands can reference the same access list. An access list can control traffic arriving at the router or leaving the router, but not traffic originating at the router. Note that, traffic such as SSH, ICMP and telnet traffic are blocked by ACL, in spite of being originated from the router. This is because, those packets are not injected as high priority packets, and hence get subjected to ACL processing. At the same time, BGP traffic bypasses the ACL applied on the interface, as it is a control packet which is injected as a critical inject packet from RSP or LC. Such packets are handled in the system with high priority and do not get dropped.

IP Access List Process and Rules

Use the following process and rules when configuring an IP access list:

- The software tests the source or destination address or the protocol of each packet being filtered against the conditions in the access list, one condition (permit or deny statement) at a time.
- If a packet does not match an access list statement, the packet is then tested against the next statement in the list.
- If a packet and an access list statement match, the remaining statements in the list are skipped and the packet is permitted or denied as specified in the matched statement. The first entry that the packet matches determines whether the software permits or denies the packet. That is, after the first match, no subsequent entries are considered.
- If the access list denies the address or protocol, the software discards the packet and returns an Internet Control Message Protocol (ICMP) Host Unreachable message. ICMP is configurable in the Cisco IOS XR software.
- If no conditions match, the software drops the packet because each access list ends with an unwritten or implicit deny statement. That is, if the packet has not been permitted or denied by the time it was tested against each statement, it is denied.
- The access list should contain at least one permit statement or else all packets are denied.
- Because the software stops testing conditions after the first match, the order of the conditions is critical. The same permit or deny statements specified in a different order could result in a packet being passed under one circumstance and denied in another circumstance.
- Only one access list per interface, per protocol, per direction is allowed.
- Inbound access lists process packets arriving at the router. Incoming packets are processed before being routed to an outbound interface. An inbound access list is efficient because it saves the overhead of routing lookups if the packet is to be discarded because it is denied by the filtering tests. If the packet is permitted by the tests, it is then processed for routing. For inbound lists, permit means continue to process the packet after receiving it on an inbound interface; **deny** means discard the packet.
- Outbound access lists process packets before they leave the router. Incoming packets are routed to the outbound interface and then processed through the outbound access list. For outbound lists, permit means send it to the output buffer; deny means discard the packet.
- An access list can not be removed if that access list is being applied by an access group in use. To remove an access list, remove the access group that is referencing the access list and then remove the access list.
- An access list must exist before you can use the **ipv4 access group** command.

Helpful Hints for Creating IP Access Lists

Consider the following when creating an IP access list:

- Create the access list before applying it to an interface. An interface to which an empty access list is applied permits all traffic.
- If you applied a nonexistent access list to an interface and then proceed to configure the access list, the first statement is placed into effect, and the the implicit deny statement that follows could cause all other traffic that needs to be permitted on the interface to be dropped, until you configure statements allowing the dropped traffic to be permitted.
- Organize your access list so that more specific references in a network or subnet appear before more general ones.

- To make the purpose of individual statements more easily understood at a glance, you can write a helpful remark before or after any statement.

Source and Destination Addresses

Source address and destination addresses are two of the most typical fields in an IP packet on which to base an access list. Specify source addresses to control packets from certain networking devices or hosts. Specify destination addresses to control packets being sent to certain networking devices or hosts.

Wildcard Mask and Implicit Wildcard Mask

Address filtering uses wildcard masking to indicate whether the software checks or ignores corresponding IP address bits when comparing the address bits in an access-list entry to a packet being submitted to the access list. By carefully setting wildcard masks, an administrator can select a single or several IP addresses for permit or deny tests.

Wildcard masking for IP address bits uses the number 1 and the number 0 to specify how the software treats the corresponding IP address bits. A wildcard mask is sometimes referred to as an *inverted mask*, because a 1 and 0 mean the opposite of what they mean in a subnet (network) mask.

- A wildcard mask bit 0 means *check* the corresponding bit value.
- A wildcard mask bit 1 means *ignore* that corresponding bit value.

You do not have to supply a wildcard mask with a source or destination address in an access list statement. If you use the **host** keyword, the software assumes a wildcard mask of 0.0.0.0.

Unlike subnet masks, which require contiguous bits indicating network and subnet to be ones, wildcard masks allow noncontiguous bits in the mask. For IPv6 access lists, only contiguous bits are supported.

You can also use CIDR format (/x) in place of wildcard bits. For example, the IPv4 address 1.2.3.4 0.255.255.255 corresponds to 1.2.3.4/8

Transport Layer Information

You can filter packets on the basis of transport layer information, such as whether the packet is a TCP, UDP, SCTP, ICMP, or IGMP packet.

IP Access List Entry Sequence Numbering

The ability to apply sequence numbers to IP access-list entries simplifies access list changes. Prior to this feature, there was no way to specify the position of an entry within an access list. If a user wanted to insert an entry (statement) in the middle of an existing list, all the entries after the desired position had to be removed, then the new entry was added, and then all the removed entries had to be reentered. This method was cumbersome and error prone.

The IP Access List Entry Sequence Numbering feature allows users to add sequence numbers to access-list entries and resequence them. When you add a new entry, you choose the sequence number so that it is in a desired position in the access list. If necessary, entries currently in the access list can be resequenced to create room to insert the new entry.

Sequence Numbering Behavior

The following details the sequence numbering behavior:

- If entries with no sequence numbers are applied, the first entry is assigned a sequence number of 10, and successive entries are incremented by 10. The maximum configurable sequence number is 2147483643 for IPv4 and IPv6 entries. For other entries, the maximum configurable sequence number is 2147483646. If the generated sequence number exceeds this maximum number, the following message displays:

Exceeded maximum sequence number.

- If you provide an entry without a sequence number, it is assigned a sequence number that is 10 greater than the last sequence number in that access list and is placed at the end of the list.
- ACL entries can be added without affecting traffic flow and hardware performance.
- If a new access list is entered from global configuration mode, then sequence numbers for that access list are generated automatically.
- Distributed support is provided so that the sequence numbers of entries in the route processor (RP) and line card (LC) are synchronized at all times.
- This feature works with named standard and extended IP access lists. Because the name of an access list can be designated as a number, numbers are acceptable.

Understanding IP Access List Logging Messages

Cisco IOS XR software can provide logging messages about packets permitted or denied by a standard IP access list. That is, any packet that matches the access list causes an informational logging message about the packet to be sent to the console. The level of messages logged to the console is controlled by the **logging console** command in global configuration mode.



Note ACL logging isn't supported for ingress MPLS packets

The first packet that triggers the access list causes an immediate logging message, and subsequent packets are collected over 5-minute intervals before they are displayed or logged.

However, you can use the { **ipv4 | ipv6** } **access-list log-update threshold** command to set the number of packets that, when they match an access list (and are permitted or denied), cause the system to generate a log message. You might do this to receive log messages more frequently than at 5-minute intervals.



Caution If you set the *update-number* argument to 1, a log message is sent right away, rather than caching it; every packet that matches an access list causes a log message. A setting of 1 isn't recommended because the volume of log messages could overwhelm the system.

Even if you use the { **ipv4 | ipv6** } **access-list log-update threshold** command, the 5-minute timer remains in effect, so each cache is emptied at the end of 5 minutes, regardless of the number of messages in each cache. Regardless of when the log message is sent, the cache is flushed and the count reset to 0 for that message the same way it's when a threshold isn't specified.



Note The logging facility might drop some logging message packets if there are too many to be handled or if more than one logging message is handled in 1 second. This behavior prevents the router from using excessive CPU cycles because of too many logging packets. Therefore, the logging facility shouldn't be used as a billing tool or as an accurate source of the number of matches to an access list.

Enable Logging on ACE

This section shows you how to enable the ACE of an ACL to log informational messages when it matches incoming packets, using the optional keyword **log**. The router supports this feature only for IPv4 or IPv6 ingress ACLs. The logging message includes the access list number, whether the packet was permitted or denied, the source IP address of the packet, and the number of packets from that source permitted or denied in the prior 5-minute interval.

```
Router#configure
Router(config)#ipv4 access-list test
Router(config-ipv4-acl)#10 permit udp 10.85.1.0 255.255.255.0 log
Router(config-ipv4-acl)#exit
Router(config)# interface FortyGigE0/0/0/22
Router(config-if)# ipv4 access-group test ingress
Router(config-if)# commit
```



Note Set log-level to **informational** or higher with the **logging console** command, so that the router displays the ACL log-messages on the console.

```
Router#configure
Router(config)#logging console informational
Router(config)# commit
```

For more information on log-levels, see section *Syslog Message Severity Levels* in the *Implementing System Logging* chapter of the *System Monitoring Configuration Guide for Cisco NCS 5500 Series Routers*.

The following snippet shows a sample log message:

```
Router: ipv4_acl_mgr[350]: %ACL-IPV4_ACL-6-IPACCESSLOGP : access-list test (10) permit udp
10.85.1.2(0) -> 10.0.0.1(0), 1 packet
```

Extended Access Lists with Fragment Control

In earlier releases, the non-fragmented packets and the initial fragments of a packet were processed by IP extended access lists (if you apply this access list), but non-initial fragments were permitted, by default. However, now, the IP Extended Access Lists with Fragment Control feature allows more granularity of control over non-initial fragments of a packet. Using this feature, you can specify whether the system examines non-initial IP fragments of packets when applying an IP extended access list.

As non-initial fragments contain only Layer 3 information, these access-list entries containing only Layer 3 information, can now be applied to non-initial fragments also. The fragment has all the information the system requires to filter, so the access-list entry is applied to the fragments of a packet.

This feature adds the optional **fragments** keyword to the following IP access list commands: **deny (IPv4)**, **permit (IPv4)**, **deny (IPv6)**, **permit (IPv6)**. By specifying the **fragments** keyword in an access-list entry,

that particular access-list entry applies only to non-initial fragments of packets; the fragment is either permitted or denied accordingly.

The behavior of access-list entries regarding the presence or absence of the **fragments** keyword can be summarized as follows:

| If the Access-List Entry has... | Then... |
|---|---|
| ...no fragments keyword and all of the access-list entry information matches, | <p>For an access-list entry containing only Layer 3 information:</p> <ul style="list-style-type: none"> • The entry is applied to non-fragmented packets, initial fragments, and non-initial fragments. <p>For an access-list entry containing Layer 3 and Layer 4 information:</p> <ul style="list-style-type: none"> • The entry is applied to non-fragmented packets and initial fragments. <ul style="list-style-type: none"> • If the entry matches and is a permit statement, the packet or fragment is permitted. • If the entry matches and is a deny statement, the packet or fragment is denied. • The entry is also applied to non-initial fragments in the following manner. Because non-initial fragments contain only Layer 3 information, only the Layer 3 portion of an access-list entry can be applied. If the Layer 3 portion of the access-list entry matches, and <ul style="list-style-type: none"> • If the entry is a permit statement, the non-initial fragment is permitted. • If the entry is a deny statement, the next access-list entry is processed. <p>Note Note that the deny statements are handled differently for non-initial fragments versus non-fragmented or initial fragments.</p> |
| ...the fragments keyword and all of the access-list entry information matches, | <p>The access-list entry is applied only to non-initial fragments.</p> <p>Note The fragments keyword cannot be configured for an access-list entry that contains any Layer 4 information.</p> |

You should not add the **fragments** keyword to every access-list entry, because the first fragment of the IP packet is considered a non-fragment and is treated independently of the subsequent fragments. Because an initial fragment will not match an access list permit or deny entry that contains the **fragments** keyword, the packet is compared to the next access list entry until it is either permitted or denied by an access list entry that does not contain the **fragments** keyword. Therefore, you may need two access list entries for every deny entry. The first deny entry of the pair will not include the **fragments** keyword, and applies to the initial fragment. The second deny entry of the pair will include the **fragments** keyword and applies to the subsequent fragments. In the cases where there are multiple **deny** access list entries for the same host but with different Layer 4 ports, a single deny access-list entry with the **fragments** keyword for that host is all that has to be added. Thus all the fragments of a packet are handled in the same manner by the access list.

Packet fragments of IP datagrams are considered individual packets and each fragment counts individually as a packet in access-list accounting and access-list violation counts.



Note The **fragments** keyword cannot solve all cases involving access lists and IP fragments.



Note Within the scope of ACL processing, Layer 3 information refers to fields located within the IPv4 header; for example, source, destination, protocol. Layer 4 information refers to other data contained beyond the IPv4 header; for example, source and destination ports for TCP or UDP, flags for TCP, type and code for ICMP.

Policy Routing

Fragmentation and the fragment control feature affect policy routing if the policy routing is based on the **match ip address** command and the access list had entries that match on Layer 4 through Layer 7 information. It is possible that noninitial fragments pass the access list and are policy routed, even if the first fragment was not policy routed or the reverse.

By using the **fragments** keyword in access-list entries as described earlier, a better match between the action taken for initial and noninitial fragments can be made and it is more likely policy routing will occur as intended.

Comments About Entries in Access Lists

You can include comments (remarks) about entries in any named IP access list using the **remark** access list configuration command. The remarks make the access list easier for the network administrator to understand and scan. Each remark line is limited to 255 characters.

The remark can go before or after a **permit** or **deny** statement. You should be consistent about where you put the remark so it is clear which remark describes which **permit** or **deny** statement. For example, it would be confusing to have some remarks *before* the associated **permit** or **deny** statements and some remarks *after* the associated statements. Remarks can be sequenced.

Remember to apply the access list to an interface or terminal line after the access list is created. See the “[Applying Access Lists, on page 92](#)” section for more information.

Access Control List Counters

In Cisco IOS XR software, ACL counters are maintained both in hardware and software. Hardware counters are used for packet filtering applications such as when an access group is applied on an interface. Software counters are used by all the applications mainly involving software packet processing.

Packet filtering makes use of 64-bit hardware counters per ACE. If the same access group is applied on interfaces that are on the same line card in a given direction, the hardware counters for the ACL are shared between two interfaces.

To display the hardware counters for a given access group, use the **show access-lists ipv4** [*access-list-name* **hardware** {**ingress** | **egress**} [**interface** *type interface-path-id*] {**location** *node-id*}] command in EXEC mode.

To clear the hardware counters, use the **clear access-list ipv4** *access-list-name* [**hardware** {**ingress** | **egress**} [**interface type interface-path-id**] {**location node-id**}] command in EXEC mode.

Hardware counting is not enabled by default for IPv4 ACLs because of a small performance penalty. To enable hardware counting, use the **ipv4 access-group** *access-list-name* {**ingress** | **egress**} [**hardware-count**] command in interface configuration mode. This command can be used as desired, and counting is enabled only on the specified interface.

Software counters are updated for the packets processed in software, for example, exception packets punted to the LC CPU for processing, or ACL used by routing protocols, and so on. The counters that are maintained are an aggregate of all the software applications using that ACL. To display software-only ACL counters, use the **show access-lists ipv4** *access-list-name* [**sequence number**] command in EXEC mode.

All the above information is true for IPv6, except that hardware counting is always enabled; there is no **hardware-count** option in the IPv6 access-group command-line interface (CLI).

BGP Filtering Using Prefix Lists

Prefix lists can be used as an alternative to access lists in many BGP route filtering commands. The advantages of using prefix lists are as follows:

- Significant performance improvement in loading and route lookup of large lists.
- Incremental updates are supported.
- More user friendly CLI. The CLI for using access lists to filter BGP updates is difficult to understand and use because it uses the packet filtering format.
- Greater flexibility.

Before using a prefix list in a command, you must set up a prefix list, and you may want to assign sequence numbers to the entries in the prefix list.

How the System Filters Traffic by Prefix List

Filtering by prefix list involves matching the prefixes of routes with those listed in the prefix list. When there is a match, the route is used. More specifically, whether a prefix is permitted or denied is based upon the following rules:

- An empty prefix list permits all prefixes.
- An implicit deny is assumed if a given prefix does not match any entries of a prefix list.
- When multiple entries of a prefix list match a given prefix, the longest, most specific match is chosen.

Sequence numbers are generated automatically unless you disable this automatic generation. If you disable the automatic generation of sequence numbers, you must specify the sequence number for each entry using the *sequence-number* argument of the **permit** and **deny** commands in either IPv4 or IPv6 prefix list configuration command. Use the **no** form of the **permit** or **deny** command with the *sequence-number* argument to remove a prefix-list entry.

The **show** commands include the sequence numbers in their output.

Information About Implementing ACL-based Forwarding

To implement access lists and prefix lists, you must understand the following concepts:

ACL-based Forwarding Overview

Converged networks carry voice, video and data. Users may need to route certain traffic through specific paths instead of using the paths computed by routing protocols. This is achieved by specifying the next-hop address in ACL configurations, so that the configured next-hop address from ACL is used for forwarding packet towards its destination instead of routing packet-based destination address lookup. This feature of using next-hop in ACL configurations for forwarding is called ACL Based Forwarding (ABF).

Traffic engineering over an IP or MPLS backbone can be done without MPLS-TE. The ability to divert certain kinds of traffic on top of routing allows you to let only voice traffic travel over certain links, while allowing data traffic to be sent using regular routing.

ACL-based forwarding enables you to choose service from multiple providers for broadcast TV over IP, IP telephony, data, and so on, which provides a cafeteria-like access to the Internet. Service providers can divert user traffic to various content providers.

ACL-based Forwarding Functions

ACL-based forwarding (ABF) enables you to configure filters for IPv4 packets. Each packet is based on the information from an IP source or destination address, TCP ports, precedence, DSCP, and so on. If a match occurs, ABF forwards the packet to one of the multiple next hops (up to three). ABF provides an alternative to regular routing by giving the ability to forward a next hop, based on packet content that extends beyond the destination IP address.

The ABF rule does not apply to “For Us” packets.

By implementing ABF, you can perform the following functions:

- Specify up to three next hops in the ACL rules.
- Forward IPv4 packets that are being forwarded on default routes to the next hop, as specified by the ACL rule.
- Use the existing ACL matching functionality to pick up the next-hop IP address that is based on the ACE configuration. The highest preferred, active, next-hop IP address—which is based on the ACE configuration—is chosen.
- Use the traditional destination IP address forwarding if the ABF next hops are not reachable.
- Use ABF as an ingress-only feature; it is not available for packets switched or originated by the software.
- Specify no rejection when both VRF and ABF configurations are applied on an interface. The ABF configuration is silently ignored by the forwarding software.

How to Implement Access Lists and Prefix Lists

This section contains the following procedures:

Configuring Extended Access Lists

This task configures an extended IPv4 or IPv6 access list.

SUMMARY STEPS

1. **configure**
2. **{ipv4 | ipv6} access-list name**
3. **[sequence-number] remark remark**
4. Do one of the following:
 - **[sequence-number] {permit | deny} source source-wildcard destination destination-wildcard [precedence precedence] [dscp dscp] [fragments] [log | log-input]**
 - **[sequence-number] {permit | deny} protocol {source-ipv6-prefix/prefix-length | any | host source-ipv6-address} [operator {port | protocol-port}] {destination-ipv6-prefix/prefix-length | any | host destination-ipv6-address} [operator {port | protocol-port}] [dscp value] [routing] [authen] [destopts] [fragments] [log | log-input]**
5. Repeat Step 4 as necessary, adding statements by sequence number where you planned. Use the **no sequence-number** command to delete an entry.
6. **commit**
7. **show access-lists {ipv4 | ipv6} [access-list-name hardware {ingress | egress} [interface type interface-path-id] {sequence number | location node-id} summary [access-list-name] | access-list-name [sequence-number] | maximum [detail] [usage {pfilter location node-id}]]**

DETAILED STEPS

| | Command or Action | Purpose |
|--------|---|--|
| Step 1 | configure | |
| Step 2 | {ipv4 ipv6} access-list name Example: <pre>RP/0/RP0/CPU0:router(config)# ipv4 access-list acl_1</pre> OR <pre>RP/0/RP0/CPU0:router(config)# ipv6 access-list acl_2</pre> | Enters either IPv4 or IPv6 access list configuration mode and configures the named access list. |
| Step 3 | [sequence-number] remark remark Example: <pre>RP/0/RP0/CPU0:router(config-ipv4-acl)# 10 remark Do not allow user1 to telnet out</pre> | (Optional) Allows you to comment about a permit or deny statement in a named access list. <ul style="list-style-type: none"> • The remark can be up to 255 characters; anything longer is truncated. • Remarks can be configured before or after permit or deny statements, but their location should be consistent. |

| | Command or Action | Purpose |
|---------------|---|--|
| Step 4 | <p>Do one of the following:</p> <ul style="list-style-type: none"> [<i>sequence-number</i>] { permit deny } <i>source source-wildcard destination destination-wildcard</i> [precedence <i>precedence</i>] [dscp <i>dscp</i>] [fragments] [log log-input] [<i>sequence-number</i>] { permit deny } <i>protocol</i> { <i>source-ipv6-prefix/prefix-length</i> any host <i>source-ipv6-address</i> } [<i>operator</i> { <i>port</i> <i>protocol-port</i> }] { <i>destination-ipv6-prefix/prefix-length</i> any host <i>destination-ipv6-address</i> } [<i>operator</i> { <i>port</i> <i>protocol-port</i> }] [dscp <i>value</i>] [<i>routing</i>] [authen] [destopts] [fragments] [log log-input] <p>Example:</p> <pre>RP/0/RP0/CPU0:router(config-ipv4-acl)# 10 permit 172.16.0.0 0.0.255.255 RP/0/RP0/CPU0:router(config-ipv4-acl)# 20 deny 192.168.34.0 0.0.0.255</pre> <p>or</p> <pre>RP/0/RP0/CPU0:router(config-ipv6-acl)# 20 permit icmp any any RP/0/RP0/CPU0:router(config-ipv6-acl)# 30 deny tcp any any gt 5000</pre> | <p>Specifies one or more conditions allowed or denied in IPv4 access list <i>acl_1</i>.</p> <ul style="list-style-type: none"> The optional log keyword causes an information logging message about the packet that matches the entry to be sent to the console. The optional log-input keyword provides the same function as the log keyword, except that the logging message also includes the input interface. <p>or</p> <p>Specifies one or more conditions allowed or denied in IPv6 access list <i>acl_2</i>.</p> <ul style="list-style-type: none"> Refer to the deny (IPv6) and permit (IPv6) commands for more information on filtering IPv6 traffic based on based on IPv6 option headers and optional, upper-layer protocol type information. <p>Note Every IPv6 access list has an implicit deny ipv6 any any statement as its last match condition. An IPv6 access list must contain at least one entry for the implicit deny ipv6 any any statement to take effect.</p> |
| Step 5 | Repeat Step 4 as necessary, adding statements by sequence number where you planned. Use the no <i>sequence-number</i> command to delete an entry. | Allows you to revise an access list. |
| Step 6 | commit | |
| Step 7 | <p>show access-lists { ipv4 ipv6 } [<i>access-list-name</i> hardware { ingress egress } [interface <i>type interface-path-id</i>] { sequence number location <i>node-id</i> } summary [<i>access-list-name</i>] <i>access-list-name</i> [<i>sequence-number</i>] maximum [detail] [usage { pfilter location <i>node-id</i> }]]</p> <p>Example:</p> <pre>RP/0/RP0/CPU0:router# show access-lists ipv4 acl_1</pre> | <p>(Optional) Displays the contents of current IPv4 or IPv6 access lists.</p> <ul style="list-style-type: none"> Use the <i>access-list-name</i> argument to display the contents of a specific access list. Use the hardware, ingress or egress, and location or sequence keywords to display the access-list hardware contents and counters for all interfaces that use the specified access list in a given direction (ingress or egress). The access group for an interface must be configured using the ipv4 access-group command for access-list hardware counters to be enabled. Use the summary keyword to display a summary of all current IPv4 or IPv6 access-lists. Use the interface keyword to display interface statistics. |

What to do next

After creating an access list, you must apply it to a line or interface. See the [Applying Access Lists, on page 92](#) section for information about how to apply an access list.

ACL commit fails while adding and removing unique Access List Entries (ACE). This happens due to the absence of an assigned manager process. The user has to exit the config-ipv4-acl mode to configuration mode and re-enter the config-ipv4-acl mode before adding the first ACE.

Applying Access Lists

After you create an access list, you must reference the access list to make it work. Access lists can be applied on *either* outbound or inbound interfaces. This section describes guidelines on how to accomplish this task for both terminal lines and network interfaces.

Set identical restrictions on all the virtual terminal lines, because a user can attempt to connect to any of them.

For inbound access lists, after receiving a packet, Cisco IOS XR software checks the source address of the packet against the access list. If the access list permits the address, the software continues to process the packet. If the access list rejects the address, the software discards the packet and returns an ICMP host unreachable message. The ICMP message is configurable.

For outbound access lists, after receiving and routing a packet to a controlled interface, the software checks the source address of the packet against the access list. If the access list permits the address, the software sends the packet. If the access list rejects the address, the software discards the packet and returns an ICMP host unreachable message.

When you apply an access list that has not yet been defined to an interface, the software acts as if the access list has not been applied to the interface and accepts all packets. Note this behavior if you use undefined access lists as a means of security in your network.

Controlling Access to an Interface

This task applies an access list to an interface to restrict access to that interface.

Access lists can be applied on *either* outbound or inbound interfaces.

SUMMARY STEPS

1. **configure**
2. **interface** *type interface-path-id*
3. Do one of the following:
 - **ipv4 access-group** *access-list-name* {**ingress** | **egress**} [**hardware-count**] [**interface-statistics**]
 - **ipv6 access-group** *access-list-name* {**ingress** | **egress**} [**interface-statistics**]
4. **commit**

DETAILED STEPS

| | Command or Action | Purpose |
|--------|-------------------|---------|
| Step 1 | configure | |

| | Command or Action | Purpose |
|--------|---|--|
| Step 2 | <p>interface <i>type interface-path-id</i></p> <p>Example:</p> <pre>RP/0/RP0/CPU0:router(config)# interface GigabitEthernet 0/2/0/2</pre> | <p>Configures an interface and enters interface configuration mode.</p> <ul style="list-style-type: none"> The <i>type</i> argument specifies an interface type. For more information on interface types, use the question mark (?) online help function. The <i>instance</i> argument specifies either a physical interface instance or a virtual instance. <ul style="list-style-type: none"> The naming notation for a physical interface instance is <i>rack/slot/module/port</i>. The slash (/) between values is required as part of the notation. The number range for a virtual interface instance varies depending on the interface type. |
| Step 3 | <p>Do one of the following:</p> <ul style="list-style-type: none"> ipv4 access-group <i>access-list-name</i> {ingress egress} [hardware-count] [interface-statistics] ipv6 access-group <i>access-list-name</i> {ingress egress} [interface-statistics] <p>Example:</p> <pre>RP/0/RP0/CPU0:router(config-if)# ipv4 access-group p-in-filter in RP/0/RP0/CPU0:router(config-if)# ipv4 access-group p-out-filter out</pre> | <p>Controls access to an interface.</p> <ul style="list-style-type: none"> Use the <i>access-list-name</i> argument to specify a particular IPv4 or IPv6 access list. Use the in keyword to filter on inbound packets or the out keyword to filter on outbound packets. Use the hardware-count keyword to enable hardware counters for the IPv4 access group. <ul style="list-style-type: none"> Hardware counters are automatically enabled for IPv6 access groups. Use the interface-statistics keyword to specify per-interface statistics in the hardware. <p>This example applies filters on packets inbound and outbound from GigabitEthernet interface 0/2/0/2.</p> |
| Step 4 | commit | |

Controlling Access to a Line

This task applies an access list to a line to control access to that line.

SUMMARY STEPS

- configure**
- line** {**aux** | **console** | **default** | **template** *template-name*}
- access-class** *list-name* {**ingress** | **egress**}
- commit**

DETAILED STEPS

| | Command or Action | Purpose |
|--------|--|--|
| Step 1 | configure | |
| Step 2 | line {aux console default template <i>template-name</i> } Example: <pre>RP/0/RP0/CPU0:router(config)# line default</pre> | <p>Specifies either the auxiliary, console, default, or a user-defined line template and enters line template configuration mode.</p> <ul style="list-style-type: none"> Line templates are a collection of attributes used to configure and manage physical terminal line connections (the console and auxiliary ports) and vty connections. The following templates are available in Cisco IOS XR software: <ul style="list-style-type: none"> Aux line template—The line template that applies to the auxiliary line. Console line template— The line template that applies to the console line. Default line template—The default line template that applies to a physical and virtual terminal lines. User-defined line templates—User-defined line templates that can be applied to a range of virtual terminal lines. |
| Step 3 | access-class <i>list-name</i> {ingress egress} Example: <pre>RP/0/RP0/CPU0:router(config-line)# access-class acl_2 out</pre> | <p>Restricts incoming and outgoing connections using an IPv4 or IPv6 access list.</p> <ul style="list-style-type: none"> In the example, outgoing connections for the default line template are filtered using the IPv6 access list <code>acl_2</code>. |
| Step 4 | commit | |

Configuring Prefix Lists

This task configures an IPv4 or IPv6 prefix list.

SUMMARY STEPS

- configure**
- {**ipv4** | **ipv6**} **prefix-list** *name*
- [*sequence-number*] **remark** *remark*
- [*sequence-number*] {**permit** | **deny**} *network/length* [**ge** *value*] [**le** *value*] [**eq** *value*]
- Repeat Step 4 as necessary. Use the **no** *sequence-number* command to delete an entry.
- commit**
- Do one of the following:

- **show prefix-list ipv4** [*name*] [*sequence-number*]
- **show prefix-list ipv6** [*name*] [*sequence-number*] [*summary*]

8. **clear** {*ipv4* | *ipv6*} **prefix-list** *name* [*sequence-number*]

DETAILED STEPS

| | Command or Action | Purpose |
|---------------|---|--|
| Step 1 | configure | |
| Step 2 | <p>{<i>ipv4</i> <i>ipv6</i>} prefix-list <i>name</i></p> <p>Example:</p> <pre>RP/0/RP0/CPU0:router(config)# ipv4 prefix-list pfx_1</pre> <p>or</p> <pre>RP/0/RP0/CPU0:router(config)# ipv6 prefix-list pfx_2</pre> | <p>Enters either IPv4 or IPv6 prefix list configuration mode and configures the named prefix list.</p> <ul style="list-style-type: none"> • To create a prefix list, you must enter at least one permit or deny clause. • Use the no {<i>ipv4</i> <i>ipv6</i>} prefix-list <i>name</i> command to remove all entries in a prefix list. |
| Step 3 | <p>[<i>sequence-number</i>] remark <i>remark</i></p> <p>Example:</p> <pre>RP/0/RP0/CPU0:router(config-ipv4_pfx)# 10 remark Deny all routes with a prefix of 10/8</pre> <pre>RP/0/RP0/CPU0:router(config-ipv4_pfx)# 20 deny 10.0.0.0/8 le 32</pre> | <p>(Optional) Allows you to comment about the following permit or deny statement in a named prefix list.</p> <ul style="list-style-type: none"> • The remark can be up to 255 characters; anything longer is truncated. • Remarks can be configured before or after permit or deny statements, but their location should be consistent. |
| Step 4 | <p>[<i>sequence-number</i>] {permit deny} <i>network/length</i> [<i>ge value</i>] [<i>le value</i>] [<i>eq value</i>]</p> <p>Example:</p> <pre>RP/0/RP0/CPU0:router(config-ipv6_pfx)# 20 deny 128.0.0.0/8 eq 24</pre> | <p>Specifies one or more conditions allowed or denied in the named prefix list.</p> <ul style="list-style-type: none"> • This example denies all prefixes matching /24 in 128.0.0.0/8 in prefix list pfx_2. |
| Step 5 | Repeat Step 4 as necessary. Use the no <i>sequence-number</i> command to delete an entry. | Allows you to revise a prefix list. |
| Step 6 | commit | |
| Step 7 | <p>Do one of the following:</p> <ul style="list-style-type: none"> • show prefix-list ipv4 [<i>name</i>] [<i>sequence-number</i>] • show prefix-list ipv6 [<i>name</i>] [<i>sequence-number</i>] [<i>summary</i>] <p>Example:</p> <pre>RP/0/RP0/CPU0:router# show prefix-list ipv4 pfx_1</pre> <p>or</p> | <p>(Optional) Displays the contents of current IPv4 or IPv6 prefix lists.</p> <ul style="list-style-type: none"> • Use the <i>name</i> argument to display the contents of a specific prefix list. • Use the <i>sequence-number</i> argument to specify the sequence number of the prefix-list entry. • Use the summary keyword to display summary output of prefix-list contents. |

| | Command or Action | Purpose |
|---------------|---|--|
| | RP/0/RP0/CPU0:router# show prefix-list ipv6 pfx_2 summary | |
| Step 8 | clear {ipv4 ipv6} prefix-list name [sequence-number] Example: RP/0/RP0/CPU0:router# clear prefix-list ipv4 pfx_1 30 | (Optional) Clears the hit count on an IPv4 or IPv6 prefix list. Note The <i>hit count</i> is a value indicating the number of matches to a specific prefix-list entry. |

Configuring Standard Access Lists

This task configures a standard IPv4 access list.

Standard access lists use source addresses for matching operations.

SUMMARY STEPS

1. **configure**
2. **ipv4 access-list name**
3. [*sequence-number*] **remark remark**
4. [*sequence-number*] { **permit** | **deny** } *source* [*source-wildcard*] [**log** | **log-input**]
5. Repeat Step 4 as necessary, adding statements by sequence number where you planned. Use the **no sequence-number** command to delete an entry.
6. **commit**
7. **show access-lists [ipv4 | ipv6] [access-list-name hardware {ingress | egress} [interface type interface-path-id] {sequence number | location node-id} summary [access-list-name] | access-list-name [sequence-number] | maximum [detail] [usage {pfilter location node-id}]]**

DETAILED STEPS

| | Command or Action | Purpose |
|---------------|---|--|
| Step 1 | configure | |
| Step 2 | ipv4 access-list name Example: RP/0/RP0/CPU0:router# ipv4 access-list acl_1 | Enters IPv4 access list configuration mode and configures access list acl_1. |
| Step 3 | [<i>sequence-number</i>] remark remark Example: RP/0/RP0/CPU0:router(config-ipv4-acl)# 10 remark Do not allow user1 to telnet out | (Optional) Allows you to comment about the following permit or deny statement in a named access list. <ul style="list-style-type: none"> • The remark can be up to 255 characters; anything longer is truncated. • Remarks can be configured before or after permit or deny statements, but their location should be consistent. |

| | Command or Action | Purpose |
|--------|--|---|
| Step 4 | <p>[<i>sequence-number</i>] { permit deny } <i>source</i> [<i>source-wildcard</i>] [log log-input]</p> <p>Example:</p> <pre>RP/0/RP0/CPU0:router(config-ipv4-acl)# 20 permit 172.16.0.0 0.0.255.255</pre> <p>or</p> <pre>RRP/0/RP0/CPU0:routerrouter(config-ipv4-acl)# 30 deny 192.168.34.0 0.0.0.255</pre> | <p>Specifies one or more conditions allowed or denied, which determines whether the packet is passed or dropped.</p> <ul style="list-style-type: none"> • Use the <i>source</i> argument to specify the number of network or host from which the packet is being sent. • Use the optional <i>source-wildcard</i> argument to specify the wildcard bits to be applied to the source. • The optional log keyword causes an information logging message about the packet that matches the entry to be sent to the console. • The optional log-input keyword provides the same function as the log keyword, except that the logging message also includes the input interface. |
| Step 5 | Repeat Step 4 as necessary, adding statements by sequence number where you planned. Use the no sequence-number command to delete an entry. | Allows you to revise an access list. |
| Step 6 | commit | |
| Step 7 | <p>show access-lists [ipv4 ipv6] [<i>access-list-name</i> hardware { ingress egress } [interface type interface-path-id] { sequence number location node-id } summary [<i>access-list-name</i>] <i>access-list-name</i> [<i>sequence-number</i>] maximum [detail] [usage { pfilter location node-id }]]</p> <p>Example:</p> <pre>RP/0/RP0/CPU0:router# show access-lists ipv4 acl_1</pre> | <p>(Optional) Displays the contents of the named IPv4 access list.</p> <ul style="list-style-type: none"> • The contents of an IPv4 standard access list are displayed in extended access-list format. |

What to do next

After creating a standard access list, you must apply it to a line or interface. See the “[Applying Access Lists, on page 92](#)” section for information about how to apply an access list.

Copying Access Lists

This task copies an IPv4 or IPv6 access list.

SUMMARY STEPS

1. **copy access-list** { **ipv4** | **ipv6** } *source-acl* *destination-acl*
2. **show access-lists** { **ipv4** | **ipv6** } [*access-list-name* **hardware** { **ingress** | **egress** } [**interface type interface-path-id**] { **sequence number** | **location node-id** } | **summary** [*access-list-name*] | *access-list-name* [*sequence-number*] | **maximum** [**detail**] [**usage** { **pfilter location node-id** }]]

DETAILED STEPS

| | Command or Action | Purpose |
|---------------|---|--|
| Step 1 | <p>copy access-list {ipv4 ipv6} <i>source-acl destination-acl</i></p> <p>Example:</p> <pre>RP/0/RP0/CPU0:router# copy ipv6 access-list list-1 list-2</pre> | <p>Creates a copy of an existing IPv4 or IPv6 access list.</p> <ul style="list-style-type: none"> • Use the <i>source-acl</i> argument to specify the name of the access list to be copied. • Use the <i>destination-acl</i> argument to specify where to copy the contents of the source access list. <ul style="list-style-type: none"> • The <i>destination-acl</i> argument must be a unique name; if the <i>destination-acl</i> argument name exists for an access list, the access list is not copied. |
| Step 2 | <p>show access-lists {ipv4 ipv6} [<i>access-list-name hardware</i> {ingress egress} [interface type interface-path-id] {sequence number location node-id} summary [<i>access-list-name</i>] <i>access-list-name [sequence-number]</i> maximum [detail] [usage {pfilter location node-id}]]</p> <p>Example:</p> <pre>RP/0/RP0/CPU0:router# show access-lists ipv4 list-2</pre> | <p>(Optional) Displays the contents of a named IPv4 or IPv6 access list. For example, you can verify the output to see that the destination access list list-2 contains all the information from the source access list list-1.</p> |

Sequencing Access-List Entries and Revising the Access List

This task shows how to assign sequence numbers to entries in a named access list and how to add or delete an entry to or from an access list. It is assumed that a user wants to revise an access list. Resequencing an access list is optional.

SUMMARY STEPS

1. **resequence access-list** {**ipv4** | **ipv6**} *name [base [increment]]*
2. **configure**
3. {**ipv4** | **ipv6**} **access-list** *name*
4. Do one of the following:
 - [*sequence-number*] {**permit** | **deny**} *source source-wildcard destination destination-wildcard [precedence precedence] [dscp dscp] [fragments] [log | log-input]*
 - [*sequence-number*] {**permit** | **deny**} *protocol {source-ipv6-prefix/prefix-length | any | host source-ipv6-address} [operator {port | protocol-port}] {destination-ipv6-prefix/prefix-length | any | host destination-ipv6-address} [operator {port | protocol-port}] [dscp value] [routing] [authen] [destopts] [fragments] [log | log-input]*
5. Repeat Step 4 as necessary, adding statements by sequence number where you planned. Use the **no sequence-number** command to delete an entry.
6. **commit**

7. **show access-lists** [ipv4 | ipv6] [access-list-name hardware {ingress | egress} [interface type interface-path-id] {sequence number | location node-id} | summary [access-list-name] | access-list-name [sequence-number] | maximum [detail] [usage {pfilter location node-id}]]

DETAILED STEPS

| | Command or Action | Purpose |
|--------|---|---|
| Step 1 | <p>resequence access-list {ipv4 ipv6} name [base [increment]]</p> <p>Example:</p> <pre>RP/0/RP0/CPU0:router# resequence access-list ipv4 acl_3 20 15</pre> | <p>(Optional) Resequences the specified IPv4 or IPv6 access list using the starting sequence number and the increment of sequence numbers.</p> <ul style="list-style-type: none"> This example resequences an IPv4 access list named <code>acl_3</code>. The starting sequence number is 20 and the increment is 15. If you do not select an increment, the default increment 10 is used. |
| Step 2 | configure | |
| Step 3 | <p>{ipv4 ipv6} access-list name</p> <p>Example:</p> <pre>RP/0/RP0/CPU0:router(config)# ipv4 access-list acl_1</pre> <p>or</p> <pre>RP/0/RP0/CPU0:router(config)# ipv6 access-list acl_2</pre> | <p>Enters either IPv4 or IPv6 access list configuration mode and configures the named access list.</p> |
| Step 4 | <p>Do one of the following:</p> <ul style="list-style-type: none"> [sequence-number] { permit deny } source source-wildcard destination destination-wildcard [precedence precedence] [dscp dscp] [fragments] [log log-input] [sequence-number] { permit deny } protocol {source-ipv6-prefix/prefix-length any host source-ipv6-address} [operator {port protocol-port}] {destination-ipv6-prefix/prefix-length any host destination-ipv6-address} [operator {port protocol-port}] [dscp value] [routing] [authen] [destopts] [fragments] [log log-input] <p>Example:</p> <pre>RP/0/RP0/CPU0:router(config-ipv4-acl)# 10 permit 172.16.0.0 0.0.255.255 RP/0/RP0/CPU0:router(config-ipv4-acl)# 20 deny 192.168.34.0 0.0.0.255</pre> <p>or</p> <pre>RP/0/RP0/CPU0:router(config-ipv6-acl)# 20 permit icmp any any</pre> | <p>Specifies one or more conditions allowed or denied in IPv4 access list <code>acl_1</code>.</p> <ul style="list-style-type: none"> The optional log keyword causes an information logging message about the packet that matches the entry to be sent to the console. The optional log-input keyword provides the same function as the log keyword, except that the logging message also includes the input interface. This access list happens to use a permit statement first, but a deny statement could appear first, depending on the order of statements you need. <p>or</p> <p>Specifies one or more conditions allowed or denied in IPv6 access list <code>acl_2</code>.</p> <ul style="list-style-type: none"> Refer to the permit (IPv6) and deny (IPv6) commands for more information on filtering IPv6 traffic based on IPv6 option headers and upper-layer protocols such as ICMP, TCP, and UDP. |

| | Command or Action | Purpose |
|---------------|--|--|
| | RP/0/RP0/CPU0:router(config-ipv6-acl)# 30 deny tcp any any gt 5000 | Note Every IPv6 access list has an implicit deny ipv6 any any statement as its last match condition. An IPv6 access list must contain at least one entry for the implicit deny ipv6 any any statement to take effect. |
| Step 5 | Repeat Step 4 as necessary, adding statements by sequence number where you planned. Use the no sequence-number command to delete an entry. | Allows you to revise the access list. |
| Step 6 | commit | |
| Step 7 | show access-lists [ipv4 ipv6] [access-list-name hardware {ingress egress} [interface type interface-path-id] {sequence number location node-id} summary [access-list-name] access-list-name [sequence-number] maximum [detail] [usage {pfilter location node-id}]] Example: RP/0/RP0/CPU0:router# show access-lists ipv4 acl_1 | (Optional) Displays the contents of a named IPv4 or IPv6 access list. <ul style="list-style-type: none"> Review the output to see that the access list includes the updated information. |

What to do next

If your access list is not already applied to an interface or line or otherwise referenced, apply the access list. See the “[Applying Access Lists, on page 92](#)” section for information about how to apply an access list.

Copying Prefix Lists

This task copies an IPv4 or IPv6 prefix list.

SUMMARY STEPS

1. **copy prefix-list {ipv4 | ipv6} source-name destination-name**
2. Do one of the following:
 - **show prefix-list ipv4 [name] [sequence-number] [summary]**
 - **show prefix-list ipv6 [name] [sequence-number] [summary]**

DETAILED STEPS

| | Command or Action | Purpose |
|---------------|--|---|
| Step 1 | copy prefix-list {ipv4 ipv6} source-name destination-name Example: RP/0/RP0/CPU0:router# copy prefix-list ipv6 list_1 list_2 | Creates a copy of an existing IPv4 or IPv6 prefix list. <ul style="list-style-type: none"> Use the <i>source-name</i> argument to specify the name of the prefix list to be copied and the <i>destination-name</i> argument to specify where to copy the contents of the source prefix list. |

| | Command or Action | Purpose |
|---------------|--|---|
| | | <ul style="list-style-type: none"> The <i>destination-name</i> argument must be a unique name; if the <i>destination-name</i> argument name exists for a prefix list, the prefix list is not copied. |
| Step 2 | Do one of the following: <ul style="list-style-type: none"> show prefix-list ipv4 <i>[name]</i> <i>[sequence-number]</i> <i>[summary]</i> show prefix-list ipv6 <i>[name]</i> <i>[sequence-number]</i> <i>[summary]</i> Example: <pre>RP/0/RP0/CPU0:router# show prefix-list ipv6 list_2</pre> | (Optional) Displays the contents of current IPv4 or IPv6 prefix lists. <ul style="list-style-type: none"> Review the output to see that prefix list list_2 includes the entries from list_1. |

Sequencing Prefix List Entries and Revising the Prefix List

This task shows how to assign sequence numbers to entries in a named prefix list and how to add or delete an entry to or from a prefix list. It is assumed a user wants to revise a prefix list. Resequencing a prefix list is optional.

Before you begin



Note Resequencing IPv6 prefix lists is not supported.

SUMMARY STEPS

- resequence prefix-list ipv4** *name* *[base [increment]]*
- configure**
- {ipv4 | ipv6} prefix-list** *name*
- [sequence-number]* **{permit | deny}** *network/length* **[ge value]** **[le value]** **[eq value]**
- Repeat Step 4 as necessary, adding statements by sequence number where you planned. Use the **no sequence-number** command to delete an entry.
- commit**
- Do one of the following:
 - show prefix-list ipv4** *[name]* *[sequence-number]*
 - show prefix-list ipv6** *[name]* *[sequence-number]* **[summary]**

DETAILED STEPS

| | Command or Action | Purpose |
|---------------|--|---|
| Step 1 | resequence prefix-list ipv4 <i>name</i> [<i>base</i> [<i>increment</i>]] Example: <pre>RP/0/RP0/CPU0:router# resequence prefix-list ipv4 pfx_1 10 15</pre> | (Optional) Resequences the named IPv4 prefix list using the starting sequence number and the increment of sequence numbers. <ul style="list-style-type: none"> This example resequences a prefix list named pfx_1. The starting sequence number is 10 and the increment is 15. |
| Step 2 | configure | |
| Step 3 | {ipv4 ipv6} prefix-list <i>name</i> Example: <pre>RP/0/RP0/CPU0:router(config)# ipv6 prefix-list pfx_2</pre> | Enters either IPv4 or IPv6 prefix list configuration mode and configures the named prefix list. |
| Step 4 | [<i>sequence-number</i>] { permit deny } <i>network/length</i> [ge <i>value</i>] [le <i>value</i>] [eq <i>value</i>] Example: <pre>RP/0/RP0/CPU0:router(config-ipv6_pfx)# 15 deny 128.0.0.0/8 eq 24</pre> | Specifies one or more conditions allowed or denied in the named prefix list. |
| Step 5 | Repeat Step 4 as necessary, adding statements by sequence number where you planned. Use the no sequence-number command to delete an entry. | Allows you to revise the prefix list. |
| Step 6 | commit | |
| Step 7 | Do one of the following: <ul style="list-style-type: none"> show prefix-list ipv4 [<i>name</i>] [<i>sequence-number</i>] show prefix-list ipv6 [<i>name</i>] [<i>sequence-number</i>] [summary] Example: <pre>RP/0/RP0/CPU0:router# show prefix-list ipv6 pfx_2</pre> | (Optional) Displays the contents of current IPv4 or IPv6 prefix lists. <ul style="list-style-type: none"> Review the output to see that prefix list pfx_2 includes all new information. |

How to Implement ACL-based Forwarding

This section contains the following procedures:

Configuring ACL-based Forwarding with Security ACL

Perform this task to configure ACL-based forwarding with security ACL.

SUMMARY STEPS

1. **configure**
2. **ipv4 access-list** *name*
3. [*sequence-number*] **permit** *protocol source source-wildcard destination destination-wildcard* [**precedence** *precedence*] [[**default**] **nexthop1** [**vrf** *vrf-name*] **ipv4** *ipv4-address1*] **nexthop2** [**vrf** *vrf-name*] **ipv4** *ipv4-address2*] **nexthop3** [**vrf** *vrf-name*] **ipv4** *ipv4-address3*]] [**dscp** *dscp*] [**fragments**] [**log** | **log-input**] [**ttl** *ttl* [*value1* ... *value2*]]
4. **commit**
5. **show access-list ipv4** [[*access-list-name* **hardware** {**ingress** | **egress**}] [**interface** *type interface-path-id*] {**sequence number** | **location** *node-id*} | **summary** [*access-list-name*] | *access-list-name* [*sequence-number*] | **maximum** [**detail**] [**usage** {**pfilter** **location** *node-id*}]]

DETAILED STEPS

| | Command or Action | Purpose |
|--------|--|--|
| Step 1 | configure | |
| Step 2 | ipv4 access-list <i>name</i> Example: RP/0/RP0/CPU0:router(config)# ipv4 access-list security-abf-acl | Enters IPv4 access list configuration mode and configures the specified access list. |
| Step 3 | [<i>sequence-number</i>] permit <i>protocol source source-wildcard destination destination-wildcard</i> [precedence <i>precedence</i>] [[default] nexthop1 [vrf <i>vrf-name</i>] ipv4 <i>ipv4-address1</i>] nexthop2 [vrf <i>vrf-name</i>] ipv4 <i>ipv4-address2</i>] nexthop3 [vrf <i>vrf-name</i>] ipv4 <i>ipv4-address3</i>]] [dscp <i>dscp</i>] [fragments] [log log-input] [ttl <i>ttl</i> [<i>value1</i> ... <i>value2</i>]] Example: Example: RP/0/RP0/CPU0:router# show access-lists ipv4 v4_acl ipv4 access-list v4_acl 10 permit ipv4 any host 172.1.1.1 nexthop1 vrf vrf_A ipv4 1.1.1.1 nexthop2 vrf vrf_B ipv4 2.2.2.2 nexthop3 vrf vrf_C ipv4 3.3.3.3 | Sets the conditions for an IPv4 access list. The configuration example shows how to configure ACL-based forwarding with security ACL. <ul style="list-style-type: none"> • The nexthop1, nexthop2, nexthop3 keywords forward the specified next hop for this entry. • If the default keyword is configured, ACL-based forwarding action is taken only if the results of the PLU lookup for the destination of the packets determine a default route; that is, no specified route is determined to the destination of the packet. |
| Step 4 | commit | |
| Step 5 | show access-list ipv4 [[<i>access-list-name</i> hardware { ingress egress }] [interface <i>type interface-path-id</i>] { sequence number location <i>node-id</i> } summary [<i>access-list-name</i>] <i>access-list-name</i> [<i>sequence-number</i>] maximum [detail] [usage { pfilter location <i>node-id</i> }]] Example: | Displays the information for ACL software. |

| | Command or Action | Purpose |
|--|--|---------|
| | <pre>RP/0/RP0/CPU0:router# show access-lists ipv4 security-abf-acl RP/0/RP0/CPU0:router# show access-lists ipv4 v4_acl ipv4 access-list v4_acl 10 permit ipv4 any host 172.1.1.1 nexthop1 vrf vrf_A ipv4 1.1.1.1 nexthop2 vrf vrf_B ipv4 2.2.2.2 nexthop3 vrf vrf_C ipv4 3.3.3.3</pre> | |

Configuring Pure ACL-Based Forwarding for IPv6 ACL

SUMMARY STEPS

1. **configure**
2. **{ipv6 } access-list name**
3. **[sequence-number] permit protocol source source-wildcard destination destination-wildcard [precedence precedence] [default nexthop [ipv6-address1] [ipv6-address2] [ipv6-address3]] [dscp dscp] [fragments] [log | log-input] [nexthop [ipv6-address1] [ipv6-address2] [ipv6-address3]] [ttl ttl value [value1 ... value2]][vrf vrf-name [ipv6-address1] [ipv6-address2] [ipv6-address3]]**
4. **commit**

DETAILED STEPS

| | Command or Action | Purpose |
|---------------|--|---|
| Step 1 | configure | |
| Step 2 | {ipv6 } access-list name Example: <pre>RP/0/RP0/CPU0:router(config)# ipv6 access-list security-abf-acl</pre> | Enters IPv6 access list configuration mode and configures the specified access list. |
| Step 3 | [sequence-number] permit protocol source source-wildcard destination destination-wildcard [precedence precedence] [default nexthop [ipv6-address1] [ipv6-address2] [ipv6-address3]] [dscp dscp] [fragments] [log log-input] [nexthop [ipv6-address1] [ipv6-address2] [ipv6-address3]] [ttl ttl value [value1 ... value2]][vrf vrf-name [ipv6-address1] [ipv6-address2] [ipv6-address3]] Example: <pre>RP/0/RP0/CPU0:router(config-ipv6-acl)# 10 permit ipv6 host 100:1:1:2:3::1 host 10:11:12::2 nexthop1</pre> | Sets the conditions for an IPv6 access list. The configuration example shows how to configure pure ACL-based forwarding for ACL. <ul style="list-style-type: none"> • The nexthop keyword forwards the specified next hop for this entry. |

| | Command or Action | Purpose |
|--------|--|---------|
| | <code>ipv6 195:1:1:200:5ff:fe00:0</code> | |
| Step 4 | <code>commit</code> | |

Configuring Pure ACL-based Forwarding for ACL

Perform this task to configure pure ACL-based forwarding for ACL.

SUMMARY STEPS

1. `configure`
2. `{ipv4} access-list name`
3. `[sequence-number] permit protocol source source-wildcard destination destination-wildcard [precedence precedence] [default nexthop [ipv4-address1] [ipv4-address2] [ipv4-address3]] [dscp dscp] [fragments] [log | log-input] [nexthop [ipv4-address1] [ipv4-address2] [ipv4-address3]] [ttl ttl value [value1 ... value2]][vrf vrf-name [ipv4-address1] [ipv4-address2] [ipv4-address3]]`
4. `commit`
5. `show access-list ipv4 [access-list-name hardware {ingress | egress} [interface type interface-path-id] {sequence number | location node-id} | summary [access-list-name] | access-list-name [sequence-number] | maximum [detail] [usage {pfilter location node-id}]`

DETAILED STEPS

| | Command or Action | Purpose |
|--------|---|--|
| Step 1 | <code>configure</code> | |
| Step 2 | <code>{ipv4} access-list name</code> Example: RP/0/RP0/CPU0:router(config)# <code>ipv4 access-list security-abf-acl</code> | Enters IPv4 access list configuration mode and configures the specified access list. |
| Step 3 | <code>[sequence-number] permit protocol source source-wildcard destination destination-wildcard [precedence precedence] [default nexthop [ipv4-address1] [ipv4-address2] [ipv4-address3]] [dscp dscp] [fragments] [log log-input] [nexthop [ipv4-address1] [ipv4-address2] [ipv4-address3]] [ttl ttl value [value1 ... value2]][vrf vrf-name [ipv4-address1] [ipv4-address2] [ipv4-address3]]</code> Example: RP/0/RP0/CPU0:router(config-ipv4-acl)# <code>10 permit ipv4 10.0.0.0 0.255.255.255 any nexthop 50.1.1.2</code> RP/0/RP0/CPU0:router(config-ipv4-acl)# <code>15 permit ipv4 30.2.1.0 0.0.0.255 any</code> RP/0/RP0/CPU0:router(config-ipv4-acl)# <code>20 permit ipv4 30.2.0.0 0.0.255.255 any nexthop 40.1.1.2</code> | Sets the conditions for an IPv4 or an IPv6 access list. The configuration example shows how to configure pure ACL-based forwarding for ACL. <ul style="list-style-type: none"> • The nexthop keyword forwards the specified next hop for this entry. |

| | Command or Action | Purpose |
|---------------|---|--|
| | RP/0/RP0/CPU0:router(config-ipv4-acl)# 25 permit ipv4 any any | |
| Step 4 | commit | |
| Step 5 | show access-list ipv4 [<i>access-list-name</i> hardware { ingress egress } [interface <i>type interface-path-id</i>] { sequence <i>number</i> location <i>node-id</i> } summary [<i>access-list-name</i>] <i>access-list-name</i> [<i>sequence-number</i>] maximum [detail] usage { pfilter location <i>node-id</i> }]] Example: RP/0/RP0/CPU0:router# show access-lists ipv4 security-abf-acl | Displays the information for ACL software. |

ACL-Chaining

ACL-Chaining also known as Multi-ACL enables customers to apply two IPv4 or IPv6 (common and interface) ACLs on an interface for packet filtering at the router. One ACL is common across multiple interfaces on the line card. This provides Ternary Content Addressable Memory(TCAM)/HW scalability.

ACL-Chaining Overview

Currently, the packet filter process (pfilter_ea) supports only one ACL to be applied per direction and per protocol on an interface. This leads to manageability issues if there are common ACL entries needed on most interfaces. Duplicate ACEs are configured for all those interfaces, and any modification to the common ACEs needs to be performed for all ACLs.

A typical ACL on the edge box for an ISP has two sets of ACEs:

- common ISP specific ACEs (ISP protected address block)
- customer/interface specific ACEs (Customer source address block)

The purpose of these address blocks is to deny access to ISP's protected infrastructure networks and anti-spoofing protection by allowing only customer source address blocks. This results in configuring unique ACL per interface and most of the ACEs being common across all the ACLs on a box. ACL provisioning and modification is very cumbersome. Any changes to the ACE impacts every customer interface. (This also wastes the HW/TCAM resources as the common ACEs are being replicated in all ACLs).

The ACL chaining feature also known as Multi-ACL allows you to configure more than one ACL that can be applied to a single interface. The goal is to separate various types of ACLs for management, and also allow you to apply both of them on the same interface, in a defined order.

Restrictions for Common ACL

The following restrictions apply while implementing Common ACL:

- Common ACL is supported in only ingress direction and for L3 interfaces only.

- The **interface-statistics** option is not available for common ACLs.
- The **hardware-count** option is available for only IPv4 ACLs.
- Only one common IPv4 and IPv6 ACL is supported on each line card.
- The common ACL option is not available for Ethernet Service (ES) ACLs.
- You can specify only common ACL or only interface ACL or both common and interface ACL in this command.
- The **compress** option is not supported for common ACLs.

Configuring an Interface to accept Common ACL

Perform this task to configure the interface to accept a common ACL along with the interface specific ACL:

SUMMARY STEPS

1. **configure**
2. **interface** *type interface-path-id*
3. { **ipv4 | ipv6** } **access-group** { **common** *access-list-name* { [*access-list-name* **ingress** [**interface-statistics**]] | **ingress** } | *access-list-name* { **ingress | egress** } [**interface-statistics**] } [**hardware-count**]
4. **commit**

DETAILED STEPS

| | Command or Action | Purpose |
|--------|--|---|
| Step 1 | configure | |
| Step 2 | interface <i>type interface-path-id</i> Example: RP/0/RP0/CPU0:router(config)# interface TenGigE 0/2/0/1 | This command configures an interface (in this case a TenGigabitEthernet interface) and enters the interface configuration mode. |
| Step 3 | { ipv4 ipv6 } access-group { common <i>access-list-name</i> { [<i>access-list-name</i> ingress [interface-statistics]] ingress } <i>access-list-name</i> { ingress egress } [interface-statistics] } [hardware-count] Example: RP/0/RP0/CPU0:router(config-if)# ipv4 access-group common acl-p acl1 ingress | Configures the interface to accept a common ACL along with the interface specific ACL. |
| Step 4 | commit | |

Configuration Examples for Implementing Access Lists and Prefix Lists

This section provides the following configuration examples:

Resequencing Entries in an Access List: Example

The following example shows access-list resequencing. The starting value in the resequenced access list is 1, and increment value is 2. The subsequent entries are ordered based on the increment values that users provide, and the range is from 1 to 2147483646.

When an entry with no sequence number is entered, by default it has a sequence number of 10 more than the last entry in the access list.

```

ipv4 access-list acl_1
10 permit ip host 10.3.3.3 host 172.16.5.34
20 permit icmp any any
30 permit tcp any host 10.3.3.3
40 permit ip host 10.4.4.4 any
60 permit ip host 172.16.2.2 host 10.3.3.12
70 permit ip host 10.3.3.3 any log
80 permit tcp host 10.3.3.3 host 10.1.2.2
100 permit ip any any

configure
ipv6 access-list acl_1
end
resequence ipv6 access-list acl_1 10 20

ipv4 access-list acl_1
10 permit ip host 10.3.3.3 host 172.16.5.34
30 permit icmp any any
50 permit tcp any host 10.3.3.3
70 permit ip host 10.4.4.4 any
90 Dynamic test permit ip any any
110 permit ip host 172.16.2.2 host 10.3.3.12
130 permit ip host 10.3.3.3 any log
150 permit tcp host 10.3.3.3 host 10.1.2.2
170 permit ip host 10.3.3.3 any
190 permit ip any any

```

Adding Entries with Sequence Numbers: Example

In the following example, a new entry is added to IPv4 access list acl_5.

```

ipv4 access-list acl_5
2 permit ipv4 host 10.4.4.2 any
5 permit ipv4 host 10.0.0.44 any
10 permit ipv4 host 10.0.0.1 any
20 permit ipv4 host 10.0.0.2 any
configure
ipv4 access-list acl_5
15 permit 10.5.5.5 0.0.0.255
end
ipv4 access-list acl_5

```

```

2 permit ipv4 host 10.4.4.2 any
5 permit ipv4 host 10.0.0.44 any
10 permit ipv4 host 10.0.0.1 any
15 permit ipv4 10.5.5.5 0.0.0.255 any
20 permit ipv4 host 10.0.0.2 any

```

Adding Entries Without Sequence Numbers: Example

The following example shows how an entry with no specified sequence number is added to the end of an access list. When an entry is added without a sequence number, it is automatically given a sequence number that puts it at the end of the access list. Because the default increment is 10, the entry will have a sequence number 10 higher than the last entry in the existing access list.

```

configure
ipv4 access-list acl_10
permit 1.1.1.1 0.0.0.255
permit 2.2.2.2 0.0.0.255
permit 3.3.3.3 0.0.0.255
end

ipv4 access-list acl_10
10 permit ip 1.1.1.0 0.0.0.255 any
20 permit ip 2.2.2.0 0.0.0.255 any
30 permit ip 3.3.3.0 0.0.0.255 any

configure
ipv4 access-list acl_10
permit 4.4.4.4 0.0.0.255
end

ipv4 access-list acl_10
10 permit ip 1.1.1.0 0.0.0.255 any
20 permit ip 2.2.2.0 0.0.0.255 any
30 permit ip 3.3.3.0 0.0.0.255 any
40 permit ip 4.4.4.0 0.0.0.255 any

```

Configuration Examples for Implementing ACL-based Forwarding

This section provides the following configuration examples:

All configuration examples include a forwarded action **nexthop** keyword. If the **default nexthop** keyword is configured, ABF action is taken only if the pointer lookup (PLU) of the destination of the packets results in hitting a default route; for example, no specific route is specified to the packet destination.

ACL with Security and ACL-based Forwarding Access Control Entry: Example

The following example shows how to configure ACL with security and an ACL-based forwarding access control entry (ACE):

```

configure
ipv4 access-list security-abf-acl
10 permit ipv4 10.0.0.0 0.255.255.255 any
15 permit ipv4 10.2.0.0 0.0.255.255 any next-hop 10.1.1.2
20 deny ipv4 10.1.0.0 0.0.255.255 any

```

```
25 permit ipv4 10.0.0.0 0.255.255.255 any
end
```



Note For ACL-based forwarding, the following command is programmed in the hardware after access list entry (ACE) 25:

```
deny ipv4 any any
```

The following methods are used to attach the ACL for both security and ACL-based forwarding ACE to an interface in ingress direction:

- Packets entering an interface with source address 10.x.x.x are forwarded using a traditional forwarding lookup.
- Packets entering an interface with source address 30.2.x.x are forwarded to next hop 40.1.1.2 (if reachable through FIB).
- Packets entering an interface with source address 30.1.x.x are dropped by security ACE 20.
- All other packets that are entering an interface are dropped by security ACL.

Pure ACL-based Forwarding for ACL Example

The following example shows how to configure pure ABF for ACL:

```
configure
  ipv4 access-list security-abf-acl
  10 permit ipv4 10.0.0.0 0.255.255.255 any next-hop 10.1.1.2
  15 permit ipv4 10.2.1.0 0.0.0.255 any
  20 permit ipv4 10.2.0.0 0.0.255.255 any next-hop 10.1.1.2
  25 permit ipv4 any any
end
```

IPv6 ACL-based Forwarding Example

The following example shows how to configure IPv6 supported ABF:

```
configure
  ipv6 access-list v6_abf
  10 permit ipv6 host 100:1:1:2:3::1 host 10:11:12::2 nexthop1 ipv6 195:1:1:1:200:5ff:fe00:0

  20 permit ipv6 host 100:1:1:2:3::1 host 10:11:12::2 nexthop1 ipv6 195:1:1:1:200:5ff:fe00:0
  nexthop2 ipv6 192:3:2:2:200:3ff:fe00:0 nexthop3 ipv6 192:4:2:2:200:3ff:fe00:0
  30 permit ipv6 host 100:1:1:2:3::1 host 10:11:12::2 nexthop1 vrf VRF1
  40 permit ipv6 host 100:1:1:2:3::1 host 10:11:12::3 nexthop1 vrf VRF1 ipv6
  192:2:2:2:200:3ff:fe00:0
  50 permit ipv6 host 100:1:1:2:3::1 host 10:11:12::2 default nexthop1 ipv6
  195:1:1:1:200:5ff:fe00:0
  60 permit ipv6 any any nexthop1 vrf VRF1 ipv6 192:2:2:2:200:3ff:fe00:0 nexthop2 vrf
  VRF1 ipv6 192:3:2:2:200:3ff:fe00:0 nexthop3 vrf VRF1 ipv6 192:4:2:2:200:3ff:fe00:0
end
```



Note For ACL-based forwarding, the following command is programmed in hardware of the ACL:F

```
deny ipv4 any any
```

Therefore, the following ACE command must be issued to let other traffic through:

```
25 permit ipv4 any any
```

The following methods are used to attach the ACL, which is used only for an ACL-based forwarding ACE, to an interface in the ingress direction:

- Packets entering an interface with source address 10.x.x.x are forwarded to next hop 10.1.1.2 (if reachable through FIB).
- Packets entering an interface with source address 10.2.1.x are forwarded using traditional forwarding lookup.
- Packets entering an interface with source address 19.2.x.x, but not in 30.2.1.x, are forwarded to next hop 10.1.1.2 (if reachable through FIB).
- All other packets entering an interface are permitted by ACE 25 and are forwarded by using a traditional forwarding lookup.
- ACE 25 ensures that all packets not matching this ACL-based forwarding ACE are forwarded and are not dropped due to an implicit deny ACE that is installed after ACE 25 by the software.

IPv6 ACL in Class Map

In Release 4.2.1, Quality of Service (Qos) features on ASR 9000 Ethernet line card and ASR 9000 Enhanced Ethernet line card are enhanced to support these:

- ASR 9000 Enhanced Ethernet LC:
 - Support on L2 and L3 interface and sub-interface
 - Support on bundle L2 and L3 interface and sub-interface
 - Support for both ingress and egress directions
 - ICMP code and type for IPv4/IPv6
- ASR 9000 Ethernet LC:
 - Support on only L3 interface and sub-interface
 - Support on L3 bundle interface and sub-interface
 - Support for both ingress and egress directions
 - ICMP code and type for IPv4/IPv6
- IPv6-supported match fields:

- IPv6 Source Address
 - IPv6 Destination Address
 - IPv6 Protocol
 - Time to live (TTL) or hop limit
 - Source Port
 - Destination Port
 - TCP Flags
 - IPv6 Flags (Routing Header(RH), Authentication Header(AH) and Destination Option Header(DH))
- Class map with IPv6 ACL that also supports:
 - IPv4 ACL
 - Discard class
 - QoS Group
 - Outer CoS
 - Inner CoS
 - Outer VLAN (ASR 9000 Enhanced Ethernet LC only)
 - Inner VLAN (ASR 9000 Enhanced Ethernet LC only)
 - match-not option
 - type of service (TOS) support
 - Policy-map with IPv6 ACL supports:
 - hierarchical class-map

Configuring IPv6 ACL QoS - An Example

This example shows how to configure IPv6 ACL QoS with IPv4 ACL and other fields :

```

ipv6 access-list aclv6
10 permit ipv6 1111:6666::2/64 1111:7777::2/64 authen
30 permit tcp host 1111:4444::2 eq 100 host 1111:5555::2 ttl eq 10
!

ipv4 access-list aclv4
10 permit ipv4 host 10.6.10.2 host 10.7.10.2
!

class-map match-any c.aclv6
match access-group ipv6 aclv6
match access-group ipv4 aclv4
match cos 1
end-class-map

```



```

!
policy-map p.aclv6
class c.aclv6
  set precedence 3
!
class class-default
!
end-policy-map
!

show qos-ea km policy p.aclv6 vmr interface tenGigE 0/1/0/6.10 hw

=====
B : type & id      E : ether type    VO : vlan outer   VI : vlan inner
Q : tos/exp/group X : Reserved      DC : discard class Fl : flags
F2: L2 flags      F4: L4 flags     SP/DP: L4 ports
T : IP TTL        D : DFS class#   L : leaf class#
Pl: Protocol      G : QoS Grp     M : V6 hdr ext.  C : VMR count
=====
policy name p.aclv6 and km format type 4
Total Egress TCAM entries: 5
|B  F2 VO  VI  Q  G  DC T  F4 Pl SP  DP  M IPv4/6 SA                               IPv4/6
DA
=====
V|3019 00 0000 0000 00 00 00 00 00 00 0000 0000 80 11116666:00000000:00000000:00000000
11117777:00000000:00000000:00000000
M|0000 FF FFFF FFFF FF FF FF FF FF FF FFFF FFFF 7F 00000000:00000000:FFFFFFFF:FFFFFFFF
00000000:00000000:FFFFFFFF:FFFFFFFF
R| C=0 03080200 000000A6 F06000FF 0000FF00 0002FF00 00FF0000 FF000000 00000000
V|3019 00 0000 0000 00 00 00 0A 01 00 0064 0000 00 11114444:00000000:00000000:00000002
11115555:00000000:00000000:00000002
M|0000 FF FFFF FFFF FF FF FF 00 FE FF 0000 FFFF FF 00000000:00000000:00000000:00000000
00000000:00000000:00000000:00000000
R| C=1 03080200 000000A6 F06000FF 0000FF00 0002FF00 00FF0000 FF000000 00000000
V|3018 00 0000 0000 00 00 00 00 00 00 0000 0000 00 0A060A02 -----
0A070A02 -----
M|0000 FF FFFF FFFF FF FF FF FF FF FFFF FFFF FF 00000000 -----
00000000 -----
R| C=2 03080200 000000A6 F06000FF 0000FF00 0002FF00 00FF0000 FF000000 00000000
V|3018 00 2000 0000 00 00 00 00 00 00 0000 0000 00 00000000:00000000:00000000:00000000
00000000:00000000:00000000:00000000
M|0003 FF 1FFF FFFF FF FF FF FF FF FFFF FFFF FF FFFFFFFF:FFFFFFFF:FFFFFFFF:FFFFFFFF
FFFFFFFF:FFFFFFFF:FFFFFFFF:FFFFFFFF
R| C=3 03080200 000000A6 F06000FF 0000FF00 0002FF00 00FF0000 FF000000 00000000
V|3018 00 0000 0000 00 00 00 00 00 00 0000 0000 00 00000000:00000000:00000000:00000000
00000000:00000000:00000000:00000000
M|0003 FF FFFF FFFF FF FF FF FF FF FFFF FFFF FF FFFFFFFF:FFFFFFFF:FFFFFFFF:FFFFFFFF
FFFFFFFF:FFFFFFFF:FFFFFFFF:FFFFFFFF
R| C=4 03000200 00010002 FF0000FF 0000FF00 0002FF00 00FF0000 FF000000 00000000

```

This example shows how to configure hierarchical policy map:

```

ipv6 access-list aclv6.p
10 permit ipv6 1111:1111::/8 2222:2222::/8

ipv6 access-list aclv6.c
10 permit ipv6 host 1111:1111::2 host 2222:2222::3

```

```
class-map match-any c.aclv6.c
match not access-group ipv6 aclv6.c
end-class-map
!
```

```
class-map match-any c.aclv6.p
match access-group ipv6 aclv6.p
end-class-map
!
```

```
policy-map child
class c.aclv6.c
  set precedence 7
!
```

```
policy-map parent
class c.aclv6.p
  service-policy child
  set precedence 1
```

```
(config)#do show qos-ea km policy parent vmr interface tenGigE 0/1/0/6 hw
```

```
=====
B : type & id      E : ether type    VO : vlan outer   VI : vlan inner
Q : tos/exp/group X : Reserved      DC : discard class Fl : flags
F2: L2 flags      F4: L4 flags     SP/DP: L4 ports
T : IP TTL        D : DFS class#   L : leaf class#
Pl: Protocol      G : QoS Grp     M : V6 hdr ext.  C : VMR count
=====

policy name parent and format type 4
Total Ingress TCAM entries: 3
|B  F2 VO  VI  Q  G  DC T  F4 Pl SP  DP  M  IPv4/6 SA                               IPv4/6
DA
=====
V|200D 00 0000 0000 00 00 00 00 00 00 0000 0000 00 11111111:00000000:00000000:00000002
22222222:00000000:00000000:00000003
M|0000 FF FFFF FFFF FF FF FF FF FF FFFF FFFF FF 00000000:00000000:00000000:00000000
00000000:00000000:00000000:00000000
R| C=0 11800200 00020000 29000000 80004100 00000000 00000000 00000000 00000000
V|200D 00 0000 0000 00 00 00 00 00 00 0000 0000 00 11000000:00000000:00000000:00000000
22000000:00000000:00000000:00000000
M|0000 FF FFFF FFFF FF FF FF FF FF FFFF FFFF FF 00FFFFFF:FFFFFFFF:FFFFFFFF:FFFFFFFF
00FFFFFF:FFFFFFFF:FFFFFFFF:FFFFFFFF
R| C=1 11800200 00010000 29000000 80004700 00000000 00000000 00000000 00000000
V|200C 00 0000 0000 00 00 00 00 00 00 0000 0000 00 00000000:00000000:00000000:00000000
00000000:00000000:00000000:00000000
M|0003 FF FFFF FFFF FF FF FF FF FF FFFF FFFF FF FFFFFFFF:FFFFFFFF:FFFFFFFF:FFFFFFFF
FFFFFFFF:FFFFFFFF:FFFFFFFF:FFFFFFFF
R| C=2 11000200 00030000 00000000 00000000 00000000 00000000 00000000 00000000
```

Configuring an Interface to accept Common ACL - Examples

This section provides configuration examples of common ACL.

This example shows how to replace an ACL configured on the interface without explicitly deleting the ACL:

```

Interface Pos0/2/0/0
ipv4 access-group common C_acl ACL1 ingress
commit
replace Interface acl ACL1 by ACL2
Interface Pos0/2/0/0
ipv4 access-group common C_acl ACL2 ingress
commit

```

This example shows how common ACL cannot be replaced on interfaces without deleting it explicitly from the interface:

```

Interface Pos0/2/0/0
ipv4 access-group common C_acl1 ACL1 ingress
commit
change the common acl to C_acl2
Interface Pos0/2/0/0
no ipv4 access-group common C_acl1 ACL1 ingress
commit
Interface Pos0/2/0/0
ipv4 access-group common C_acl2 ACL1 ingress
commit

```



Note When reconfiguring common ACL, you must ensure that no other interface on the line card is attached to the common ACL. In other words, atomic replacement of common ACL is not possible.



Note If both common ACL and interface ACL are attached to an interface and only one of the above is reconfigured on the interface, then the other will be removed automatically.

```

Interface Pos0/2/0/0
ipv4 access-group common C_acl1 ACL1 ingress
commit

Interface Pos0/2/0/0
ipv4 access-group ACL1 ingress
commit
This removes the common acl.

Interface Pos0/2/0/0
ipv4 access-group common C_acl1 ACL1 ingress
commit

Interface Pos0/2/0/0
ipv4 access-group common C_acl1 ingress
commit

```

This example shows how the interface ACL is removed:

```
Interface Pos0/2/0/0
ipv4 access-group common C_acl1 ACL1 ingress
commit
```

```
Interface Pos0/2/0/0
no ipv4 access-group common acl acl ingress
Commit
```

Additional References

The following sections provide references related to implementing access lists and prefix lists.

Related Documents

| Related Topic | Document Title |
|---|--|
| Access list commands: complete command syntax, command modes, command history, defaults, usage guidelines, and examples | <i>Access List Commands</i> module in <i>IP Addresses and Services Command Reference for Cisco CRS Routers</i> |
| Prefix list commands: complete command syntax, command modes, command history, defaults, usage guidelines, and examples | <i>Prefix List Commands</i> module in <i>IP Addresses and Services Command Reference for Cisco CRS Routers</i> |
| Terminal services commands: complete command syntax, command modes, command history, defaults, usage guidelines, and examples | <i>Terminal Services Commands</i> module in <i>System Management Command Reference for Cisco CRS Routers</i> |

Standards

| Standards | Title |
|---|-------|
| No new or modified standards are supported by this feature, and support for existing standards has not been modified by this feature. | — |

MIBs

| MIBs | MIBs Link |
|------|---|
| — | To locate and download MIBs, use the Cisco MIB Locator found at the following URL and choose a platform under the Cisco Access Products menu: https://mibs.cloudapps.cisco.com/ITDIT/MIBS/servlet/index |

RFCs

| RFCs | Title |
|---|-------|
| No new or modified RFCs are supported by this feature, and support for existing RFCs has not been modified by this feature. | — |

Technical Assistance

| Description | Link |
|---|---|
| The Cisco Technical Support website contains thousands of pages of searchable technical content, including links to products, technologies, solutions, technical tips, and tools. Registered Cisco.com users can log in from this page to access even more content. | http://www.cisco.com/techsupport |



CHAPTER 7

Implementing Video Monitoring

Configuring Video Monitoring is a four-step procedure, which includes configuring the relevant class-maps and policy maps, and binding the video monitoring policy to an interface.

- [Prerequisites for Implementing Video Monitoring, on page 119](#)
- [Information About Implementing Video Monitoring, on page 119](#)
- [Implementing Video Monitoring, on page 124](#)
- [Configuration Examples for Implementing Video Monitoring , on page 133](#)
- [Additional References, on page 138](#)

Prerequisites for Implementing Video Monitoring

- You must be in a user group associated with a task group that includes the proper task IDs. The command reference guides include the task IDs required for each command. If you suspect user group assignment is preventing you from using a command, contact your AAA administrator for assistance.
- You must install and activate packages for advanced video services. For detailed information about optional package installation, see *Cisco IOS XR Getting Started Guide for the Cisco CRS Router*.
- You must install and activate a package for the multicast routing software and enable multicast routing on the system. Video monitoring is supported on interfaces that are multicast-enabled. For detailed information about multicast routing, refer to the chapter Implementing Layer 3 Multicast Routing on Cisco CRS Routers.

Information About Implementing Video Monitoring

Video Monitoring

Poor video experience is a major cause for concern among service providers in terms of service costs and loss of revenue. To avoid the service costs of help desk time, NOC (network operation center) troubleshooting resources, and truck rolls, the capability of monitoring video traffic is essential. On Cisco IOS XR software, problems in video flows can be easily diagnosed by video monitoring.

Introduction to Video Monitoring

Packet loss is one common cause of video quality degradation. Its impact is more significant on compressed video flows. The video traffic transported through the service provider IP network is mostly compressed video – MPEG or similar encoding. Because of the way compression occurs, the traffic is extremely loss-sensitive. The video is encoded with an independent frame (I-frame) every few seconds, with subsequent frames being deltas from the I-frame. If the loss is in an I-frame, a 3 ms loss of traffic (roughly one IP packet) can result in a viewing degradation for up to 1.2 seconds.

Jitter is a key flow characteristic that requires careful buffer provisioning in the end device. The set top box (STB) that displays the media on a screen needs to decode the video in real-time. It buffers the incoming video stream so that it can decode and display the image smoothly. Large network jitter can lead to buffer underrun or overrun on the STB. Depending on how large the jitter is, this will create a visual distortion or a freezing of the video at display.

End-to-end delay in transmission is not significant for a broadcast-only application. However, as the video applications get to be more interactive, the end-to-end latency (delay) becomes a critical Quality of Experience (QoE) component. Data loss is a major contributor for poor QoE.

Three main contributors to poor QoE can be summed up as:

- Packet Loss
- Jitter
- Delay

Video Monitoring plays a very significant role in improving video quality, and thus, in enhancing the QoE. Video monitoring is implemented on the routers and enables network operators to measure and track video transport performance on a per-flow basis. The video packets flow through a router. We can use the packet headers and compute a metric that gives us a measure of the network performance impacting the quality of the video. This information from multiple routers is compared for the same flow to get a clear end-to-end picture of the video issues in the network and the affected flows.

Problems in video flows (and more generally, any streaming flow) can be diagnosed by video monitoring. The purpose of video monitoring is to detect perturbations and anomalies introduced by the network that cause a degraded QoE; that is, it measures the transport performance for streaming (video) traffic. Encoding errors, audio-video-lag, and other errors too cause poor QoE. However, these are introduced by the encoding device and not the network. These latter errors are not monitored.

Key Features Supported on Video Monitoring

Direct Measurements from Data Plane

Video monitoring plays a significant role in improving video quality and therefore enhances the QoE. Video monitoring implemented on Cisco CRS Routers enable the network operator to measure and track video transport performance on a per-flow basis in real time. In contrast to the conventional traffic monitoring solutions, (where sampled flows have to be sent to the control plane or additional hardware, such as dedicated blade on the router), video monitoring on Cisco CRS Router performs the monitoring operation on the data plane itself. This enables video monitoring to analyze forwarded packets in real time, to compute a metric that provides a measure of the network performance impacting the quality of the video.

Local Storage and Remote Access

Video monitoring measures packet loss and jitter at wire-speed, and stores collected information on the router, in order that the network operator can access it through a user interface. Furthermore, the performance metrics measured and stored on multiple routers can be accessed through standard SNMP from a remote operation center. These metrics provide a clear end-to-end picture of the video flow that can be composed and analyzed.

Proactive and Reactive Usages

Video monitoring on Cisco CRS Routers serve both reactive and proactive usage for service providers. It can be used to verify the quality of video service, before scaling up the service coverage to new customers. Also, it is a powerful tool for analysis and can be used to troubleshoot customer calls. Network operators can configure video monitoring to raise an alarm for various events such as variation in packet loss, jitter, flow rate, number of flows, and so on. Such an alarm can be configured to get triggered at any possible value or range.

Flow on Video Monitoring

Video monitoring uses four pieces of packet header fields to distinguish a unique flow - source IP address, destination IP address, source UDP port, and destination UDP port (this implies protocol ID is always UDP).

Unicast and Multicast

Video monitoring supports not only the monitoring of flows with IPv4 multicast destination address in the IP header, but also supports the monitoring of flows with unicast destination addresses.

Flow Rate Types and Protocol Layer

Video monitoring monitors CBR (constant bit rate) flows at the IP layer. In other words, video monitoring can monitor CBR-encoded media streams (for example, MPEG-2) encapsulated in UDP datagram, inside an IPv4 packet. Video monitoring allows users to configure expected packet rate at IP layer, or bit rate at media layer (along with the number and size of media packets).

Metrics

Video monitoring supports both packet loss and jitter metrics that follow MDI (media delivery index, RFC 4445) definition at the IP-UDP level. The MDI metrics are MLR (media loss rate) and DF (delay factor). Video monitoring uses MRV (media rate variation) which is an extension of MDI MLR; that is, MLR captures only loss, while MRV captures both loss and excess. Video monitoring DF is the same as MDI definition, where DF represents one nominal packet inter-arrival time in addition to the monitored MDI jitter. Along with the two key metrics, Video monitoring supports packet count, byte count, packet rate, bit rate, packet size, TTL (Time to Live) field in IP header, number of flows, raised alarms, and time stamp for various events.



Note The term MDI jitter, is used to signify the correctness of DF metric measured by Video monitoring. MDI jitter is measured by comparing the actual packet arrival time against the nominal arrival reference, while simple inter-packet jitter is measured by the time difference between two consecutive packet arrivals. The former captures the performance of CBR flow more precisely than the latter.

For Cisco CRS Router, only MRV is applicable.

High Availability Features

Video monitoring on Cisco CRS Router supports high availability at various levels. It supports process OIR (online insertion and removable), line card OIR, RP (route processor) fail over, and router reload. Configuration is persistent for all high availability scenarios. Monitored statistics data are preserved at process OIR and RP FO.

Interface Types and Direction

To activate video monitoring, you must configure video monitoring service policy on an interface. There are four types of interfaces to which you can attach the video monitoring policy; these are main interface, subinterface, ethernet bundle interface, and ethernet bundle subinterface. Video monitoring supports only layer 3 interfaces and not layer 2 interfaces. Video monitoring can be configured only on the input direction of the interface.

Flow Rate

Video monitoring supports standard definition (SD) video traffic (mostly compressed) of up to 100 Mbps flow rate. For uncompressed video streams, flow rate of max 3 Gbps is supported.

User Interface for Input

Video monitoring supports traditional CLI (command line interface) input for configuration that follows MQC (modular QoS configuration) syntax. You can configure video monitoring by configuring access control list (ACL), class map, and policy map; it can be activated by attaching the service policy to an interface. In-place policy modification is supported.



Note In place policy modification internally behaves like policy removal and attach. So, flows are newly detected on the interface.

User Interface for Output

Video monitoring offers various show and clear commands for retrieving the monitored statistics. Refer the Video Monitoring Commands on Cisco CRS Routers module in the *Multicast Command Reference for Cisco CRS Routers* for a detailed description of the video monitoring commands.

You can configure TCA (threshold crossing alert) as a part of the policy map to enable video monitoring to generate syslog message for various conditions. You can also retrieve standing alarms by using **show** command or through a SNMP pull. XML is supported by video monitoring.

Number of Class Maps and Policy Maps

To use video monitoring, you must configure class map and policy map that acts as a filter to determine which flow to monitor on the data plane. Video monitoring supports a maximum of 512 class maps per policy-map, and a maximum of 1500 class maps per system. It supports a maximum of 1500 policy maps on the system.

Video PIE Installation

Video monitoring requires video PIE installation. Depending on the RP type, the video pie name has the following version:

- hfr-video-px.pie

Video Monitoring Terminology

To implement and configure video monitoring service on Cisco CRS Routers, you must first understand video monitoring terminology and concepts.

Interval duration and interval updates

Video monitoring analyzes continuously all packets on the data plane for a time period called interval duration, which is configured by the user. Statistics are exported periodically at the end of each interval duration. These exported statistics are called interval updates. The status of a video monitoring flow and its transition is described solely in reference to these interval updates. Also, all exported video monitoring flow statistics are stored in terms of these interval updates.

The interval duration is a vital video monitoring parameter. Video monitoring configuration anchors upon interval duration for functions such as frequency of export, number of exports to store, time to delete inactive flows, and so on. All video monitoring functionalities, including raising alarm (for stopped flows and flows with performance degradation), are based on the contents of interval updates.

Video monitoring flows

A video monitoring flow is an instance of a packet stream whose header fields match the configured class map (and its associated access control list). A unique flow is local to the interface to which a video monitoring service policy is attached. A video monitoring flow is composed of a series of stored interval updates. A unique flow that is created on video monitoring after a monitoring interval is called a new flow.

Flow stop

If the router stops receiving packets on a monitored flow for one full interval update or longer, the monitored flow is considered as being stopped.

Flow resumption

When a stopped video monitoring flow resumes receiving packets, a normal interval update is exported in the next monitoring interval. A resumed flow has one or more zero intervals, followed by a normal interval update.

Flow switchover

A video monitoring flow on an ethernet bundle interface, or on an ethernet bundle sub-interface, may move from one physical member interface to another; that is, the packet stream stops flowing on one interface and starts flowing on another interface. This is defined as a flow switchover. In such a case, if both interfaces are on the same line card, video monitoring treats the pre-switchover flow and the post-switchover flow as the same flow. Otherwise, it treats them as two different flows.

Flow deletion

If a stopped video monitoring flow continues to export zero intervals for a configured timeout (in terms of the number of monitoring intervals), the flow is considered dead and is marked for deletion. The duration for which the user can control inactive flows is indicated using the timeout parameter. The actual deletion for all the marked flows takes place after some delay by the periodic sweeping function. Once deleted, all exported statistics (series of interval updates including zero intervals) are completely removed from storage.

Implementing Video Monitoring

Configuring Video Monitoring is a four-step procedure, which includes configuring the relevant class-maps and policy maps, and binding the video monitoring policy to an interface.

Creating IPv4 Access Lists

This step is similar to typical IPv4 access list creation and configuration. An example configuration of ACL for video monitoring is presented here for quick reference. For more details, refer to the *Implementing Access lists and Prefix lists* chapter of the *IP Addresses and Services Configuration Guide for Cisco CRS Routers*.

This task configures a standard IPv4 access list.

Standard access lists use source addresses for matching operations.



Note Video Monitoring policy allows **deny** statements in ACL configuration, but **deny** statements are treated as **permit**. Also, log or log-input is not supported in ACL configuration.

SUMMARY STEPS

1. **configure**
2. **ipv4 access-list** *name*
3. [*sequence-number*] **remark** *remark*
4. [*sequence-number*] **permit udp** *source* [*source-port*] **destination** [*destination-port*]
5. Repeat Step 4 as necessary, adding statements by sequence number. Use the **no** *sequence-number* command to delete an entry.
6. **commit**

DETAILED STEPS

| | Command or Action | Purpose |
|--------|--|--|
| Step 1 | configure | |
| Step 2 | ipv4 access-list <i>name</i> Example: RP/0/RP0/CPU0:router# ipv4 access-list acl_1 | Enters IPv4 access list configuration mode and configures access list acl_1. |
| Step 3 | [<i>sequence-number</i>] remark <i>remark</i> Example: RP/0/RP0/CPU0:router(config-ipv4-acl)# 10 remark Do not allow user1 to telnet out | (Optional) Allows you to comment on the permit statement that follows in the named access list. <ul style="list-style-type: none"> • The remark can be up to 255 characters; anything longer is truncated. • Remarks can be configured before or after permit statements, but their location details should be consistent. |

| | Command or Action | Purpose |
|---------------|---|---|
| Step 4 | <p>[<i>sequence-number</i>] permit udp <i>source</i> [<i>source-port</i>] destination [<i>destination-port</i>]</p> <p>Example:</p> <pre>RP/0/RP0/CPU0:router(config-ipv4-acl)# 20 permit udp 172.16.0.0/24 eq 5000 host 225.0.0.1 eq 5000</pre> | <p>Allows you to specify the source and destination ports with these conditions.</p> <ul style="list-style-type: none"> • Video monitoring supports only udp. • Use the <i>source</i> keyword to specify the network or host number from which the packet is being sent. • Use the optional <i>source-wildcard</i> argument to specify the wildcard bits to be applied to the source. • Use the <i>destination</i> keyword to specify the network or host number to which the packet is being sent. • Use the optional <i>destination-wildcard</i> argument to specify the wildcard bits to be applied to the destination. |
| Step 5 | Repeat Step 4 as necessary, adding statements by sequence number. Use the no <i>sequence-number</i> command to delete an entry. | Allows you to revise an access list. |
| Step 6 | commit | |

Configuring class-map

This task sets up the flow classifier. This may match either an individual flow, or it may be an aggregate filter matching several flows.

SUMMARY STEPS

1. **configure**
2. **class-map type traffic** *class-map-name*
3. **match access-group ipv4** *acl-name*
4. **end-class-map**
5. **commit**

DETAILED STEPS

| | Command or Action | Purpose |
|---------------|--|--|
| Step 1 | configure | |
| Step 2 | <p>class-map type traffic <i>class-map-name</i></p> <p>Example:</p> <pre>RP/0/RP0/CPU0:router(config)# class-map type traffic class1</pre> | Enters the class-map mode. The class-map type must always be entered as traffic. |

| | Command or Action | Purpose |
|---------------|---|--|
| Step 3 | match access-group ipv4 <i>acl-name</i> Example: <pre>RP/0/RP0/CPU0:router(config-cmap)# match access-group ipv4 acl1</pre> | Enter the ACL to be matched for this class. Only eight ACL's can be matched per class. |
| Step 4 | end-class-map Example: <pre>RP/0/RP0/CPU0:router(config-cmap)# end-class-map</pre> | Completes the class-map configuration. |
| Step 5 | commit | |

Configuring policy-map

The policy map for video monitoring is of the performance-traffic type. Only one level of hierarchy is supported for video monitoring policy-maps. This means that no hierarchical policy map configuration is supported for video monitoring.

The policy map configuration for video monitoring has these three parts:

- Flow parameters configuration: Specifies the different properties of the flow that are monitored such as interval duration, required history intervals, timeout, etc.
- Metric parameters configuration: Specifies the metrics that need to be calculated for the flow that are monitored.
- React parameters configuration: Specifies the parameters, based on which, alerts are generated for the flow.

The configuration hierarchy is from *policy* to *class* to *flow*. This means that all the parameters that are specified above are applied to all flows that match a particular class, in the policy-map. While specifying flow and react parameters for flows matching a given class is optional, its metric parameters is mandatory.

Configuring policy-map with metric parameters

The metric parameters in a policy map can be:

- Layer 3 packet rate or
- Media bit rate (with the number of media packet counts and size in the UDP payload specified).



Note Layer 3 packet rate and Media rate have mutually exclusive configuration commands.

The configuration for each metric parameter is described in this section.

Layer 3 packet-rate

SUMMARY STEPS

1. **configure**
2. **policy-map type** *performance-traffic policy-map-name*
3. **class type** *traffic class-name*
4. **monitor metric ip-cbr**
5. **rate layer3 packet** *packet-rate* **pps**
6. **commit**

DETAILED STEPS

| | Command or Action | Purpose |
|---------------|---|---|
| Step 1 | configure | |
| Step 2 | policy-map type <i>performance-traffic policy-map-name</i> Example: RP/0/RP0/CPU0:router(config)# policy-map type performance-traffic policy1 | Enters the policy-map mode. The policy-map type should always be entered as performance traffic. |
| Step 3 | class type <i>traffic class-name</i> Example: RP/0/RP0/CPU0:router(config-pmap)# class type traffic class-name | Enter the class-map to be matched for this policy. Multiple classes can be specified for a single policy. |
| Step 4 | monitor metric ip-cbr Example: RP/0/RP0/CPU0:router(config-pmap-c)# monitor metric ip-cbr | Enters the IP-CBR metric monitor submenu. Note Currently only ip-cbr metric monitoring is supported for video monitoring. |
| Step 5 | rate layer3 packet <i>packet-rate</i> pps Example: RP/0/RP0/CPU0:router(config-pmap-c-ipcbr)# rate layer3 packet packet-rate pps | Specifies the IP layer3 packet rate in packets per second (pps). |
| Step 6 | commit | |

Media bit-rate

The metric parameters for media bit-rate consist of the media bit rate, media packet count and packet size. The rate media option enables the user to specify the number of media payload packets (that is MPEG-2 datagrams) that is present in one UDP packet, and the size of each of such media payload. It is mandatory to specify the media bit rate. There are no defaults for packet count and packet size in Cisco IOS XR Software Release 3.9.1. These values must be configured.



Note With the media bit rate configured to 1052800 bps, media packet count to 7, and media packet size to 188 bytes, the media packet rate is 100 pps at layer 3. The calculation is: $1052800 / (7 * 188 * 8) = 100$ pps.

SUMMARY STEPS

1. **configure**
2. **policy-map type** *performance-traffic policy-map-name*
3. **class type** *traffic class-name*
4. **monitor metric ip-cbr**
5. **rate media bit -rate** {bps|kbps|mbps|gbps}
6. **media packet count in-layer3** *packet-count*
7. **media packet size** *packet-size*
8. **commit**

DETAILED STEPS

| | Command or Action | Purpose |
|---------------|---|--|
| Step 1 | configure | |
| Step 2 | policy-map type <i>performance-traffic policy-map-name</i> Example: RP/0/RP0/CPU0:router(config)# policy-map type performance-traffic policy1 | Enters the policy-map mode. The policy-map type should always be entered as performance traffic. |
| Step 3 | class type <i>traffic class-name</i> Example: RP/0/RP0/CPU0:router(config-pmap)# class type traffic class-name | Enters the class-map to be matched for this policy. Multiple classes can be specified for a single policy. |
| Step 4 | monitor metric ip-cbr Example: RP/0/RP0/CPU0:router(config- pmap-c)# monitor metric ip-cbr | Enters the IP-CBR metric monitor submenu. Note Currently only ip-cbr metric monitoring is supported for video monitoring. |
| Step 5 | rate media bit -rate {bps kbps mbps gbps} Example: RP/0/RP0/CPU0:router(config- pmap-c-ipcbr)# rate media 100 mbps | Specifies the media bit rate for the flow in bps, kbps, mbps or gbps. The configuration can be committed here. Optional parameters can also be specified. Note The default unit of media bit-rate is kbps. |
| Step 6 | media packet count in-layer3 <i>packet-count</i> Example: RP/0/RP0/CPU0:router(config- pmap-c-ipbr)# media packet count in-layer3 10 | Specifies the number of media packets for each IP payload. |

| | Command or Action | Purpose |
|--------|--|--|
| Step 7 | media packet size packet-size Example: <pre>RP/0/RP0/CPU0:router(config-pmap-c-ipcbr)# media packet size 188</pre> | Specifies the size in bytes for each media packet in the IP payload. |
| Step 8 | commit | |

Configuring policy-map with flow parameters

The flow parameters in a policy map are optional.

For video monitoring, the data plane continuously monitors the flows and the metrics that are exported at the end of every interval. The duration of this interval and the number of such intervals that need to be stored for each flow (history) can also be optionally specified by the user. You can specify these flow parameters for each flow:

- **Interval Duration:** The time interval at whose end, metrics are exported. This is specified in multiples of 5 (any value between 10 and 300 seconds). The default value is 30.
- **History:** The number of intervals containing flow information (flow ID, metrics, etc.) that needs be stored for each flow. This can be any value between 1 and 60. The default value is 10.
- **Timeout:** The timeout in multiples of interval duration after which an inactive flow is marked for deletion. This can be any value between 2 and 60. The default value is 2.
- **Max Flows per class:** The maximum number of flows that need to be monitored for each class in the policy. This can be any value between 1 and 1024. The default value is 4096.

Monitoring this parameter requires maintaining a count per class in hardware and metro hardware does not provide the functionality to read the stats counter. Due to this limitation, this parameter is not supported. If the user has specified this parameter in the policy, the maximum number of flows per class is limited to maximum number of flows per line card.

SUMMARY STEPS

1. **configure**
2. **policy-map type** *performance-traffic policy-map-name*
3. **class type** *traffic class-name*
4. monitor parameters
5. **{interval duration** *duration* | **flows** *number of flows* | **history** *intervals* | **timeout** *duration*}
6. **commit**

DETAILED STEPS

| | Command or Action | Purpose |
|--------|--|--|
| Step 1 | configure | |
| Step 2 | policy-map type <i>performance-traffic policy-map-name</i> Example: | Enters the policy-map mode. The policy-map type should always be entered as performance traffic. |

| | Command or Action | Purpose |
|---------------|--|--|
| | RP/0/RP0/CPU0:router(config)# policy-map type performance-traffic policy1 | |
| Step 3 | class type <i>traffic class-name</i> Example: RP/0/RP0/CPU0:router(config-pmap)# class type traffic class-name | Enter the class-map to be matched for this policy. Multiple classes can be specified for a single policy. |
| Step 4 | monitor parameters Example: RP/0/RP0/CPU0:router(config-pmap-c)# monitor parameters | Enters the flow monitor submodule. |
| Step 5 | {interval duration <i>duration</i> flows <i>number of flows</i> history <i>intervals</i> timeout <i>duration</i> } Example: RP/0/RP0/CPU0:router(config-pmap-c-fparam)# interval duration 10 | <ul style="list-style-type: none"> • Select the interval duration option to specify the interval duration per flow; range is 10 to 300 (must be in multiples of 5). The default value is 30. • Select the history option to specify the maximum number of interval data that will be stored per flow. It can be any value between 1 and 60. The default value is 10. • Select the timeout option to specify the timeout in multiples of the interval duration after which an inactive flow will be marked for deletion. Range is between 2 and 60. The default value is 0, indicating a static flow. • Select the flows option to specify the maximum number of flows that can be monitored per class. Range is between 1 and 1024. The default value is 1024. |
| Step 6 | commit | |

Configuring policy-map with react parameters

The react parameters in a policy map are optional.

The react parameters are a direct reference for the user to indicate the flow quality. The flow is continuously monitored, and at the end of the interval duration, the statistics are examined to determine whether the threshold specified by the user for the specific parameter has exceeded. If it has, a syslog alarm is generated on the console. Once the alarm is set, no further syslog notifications are issued for the condition.

The following react parameters are used to configure the policy-map:

- **Media Rate variation (MRV):** video monitoring reacts and generates an alarm if the MRV statistic of the flow crosses the user-specified threshold.

- **Media-Stop:** video monitoring reacts and generates an alarm if a flow stops; this is to indicate that no packets were received for the flow during one full monitoring interval.
- **Packet-Rate:** video monitoring reacts and generates an alarm if the packet rate of the flow crosses the user-specified threshold.
- **Flow-Count:** video monitoring reacts and generates an alarm if the flow count for each class crosses the user-specified threshold.

SUMMARY STEPS

1. **configure**
2. **policy-map type** *performance-traffic policy-map-name*
3. **class type** *traffic class-name*
4. **react react-id** {**mrsv** | **delay-factor** | **packet-rate** | **flow-count** | **media-stop**}
5. **threshold type** **immediate**
6. **threshold value** {**ge** | **gt** | **le** | **lt** | **range**} *limit*
7. **action** **syslog**
8. **alarm severity** {**error** | **critical** | **alert** | **emergency**}
9. **alarm type** {**discrete** | **grouped**}
10. **commit**

DETAILED STEPS

| | Command or Action | Purpose |
|--------|--|---|
| Step 1 | configure | |
| Step 2 | policy-map type <i>performance-traffic policy-map-name</i> Example: <pre>RP/0/RP0/CPU0:router(config)# policy-map type performance-traffic policyl</pre> | Enters the policy-map mode. The policy-map type should always be entered as performance traffic. |
| Step 3 | class type <i>traffic class-name</i> Example: <pre>RP/0/RP0/CPU0:router(config-pmap)# class type traffic class-name</pre> | Enter the class-map to be matched for this policy. Multiple classes can be specified for a single policy. |
| Step 4 | react react-id { mrsv delay-factor packet-rate flow-count media-stop } Example: <pre>RP/0/RP0/CPU0:router(config- pmap-c)# react 1 mrsv</pre> | Enters the react parameter configuration submenu. The react ID specified here needs to be unique for each class. Note For the media-stop react parameter, the threshold-type and threshold-value options are not applicable. For the flow-count react parameter, the alarm-type option is not applicable. |

| | Command or Action | Purpose |
|----------------|---|---|
| Step 5 | threshold type immediate Example: <pre>RP/0/RP0/CPU0:router(config- pmap-c-react)# threshold type immediate</pre> | Specifies the trigger type for the threshold. Currently, the available threshold type is immediate. |
| Step 6 | threshold value {ge gt le lt range} limit Example: <pre>RP/0/RP0/CPU0:router(config- pmap-c-react)# threshold value ge 50</pre> | Specifies the trigger value range for the threshold. |
| Step 7 | action syslog Example: <pre>RP/0/RP0/CPU0:router(config- pmap-c-react)# action syslog</pre> | The action keyword specifies the action to be taken if the threshold limit is surpassed. Currently, snmp and syslog actions are the only options available. |
| Step 8 | alarm severity {error critical alert emergency} Example: <pre>RP/0/RP0/CPU0:router(config- pmap-c-react)# alarm severity critical</pre> | Specifies the alarm severity for syslog. |
| Step 9 | alarm type {discrete grouped} Example: <pre>RP/0/RP0/CPU0:router(config- pmap-c-react)# alarm type discrete</pre> | Specifies the alarm type. Discrete alarm is raised for all the flows that exceed the threshold value. Grouped alarm is raised when a certain number or percentage of the flows exceeds the threshold value. |
| Step 10 | commit | |

Configuring service policy on an interface

The configured policy-map must be attached to an interface in ingress direction in order to enable the Video Monitoring service.

For ethernet bundle interface, service policy can be attached to only the bundle parent interface and not to the physical member interfaces. For ethernet bundle sub-interfaces, it can be attached to only sub-interfaces. For VLAN sub-interfaces, the service policy cannot be attached to the main interface.

SUMMARY STEPS

1. **configure**
2. **interface type interface-path-id**

3. **service-policy type performance-traffic input** *policy-name*
4. **commit**

DETAILED STEPS

| | Command or Action | Purpose |
|--------|---|--|
| Step 1 | configure | |
| Step 2 | interface <i>type interface-path-id</i> Example: <pre>RP/0/RP0/CPU0:router(config)# interface type interface-path-id</pre> | Configures an interface and enters interface configuration mode. <ul style="list-style-type: none"> • The type argument specifies an interface type. For more information on interface types, use the question mark (?) online help function. • The instance argument specifies either a physical interface instance or a virtual instance. • The naming notation for a physical interface instance is rack/slot/module/port. The slash (/) between values is required as part of the notation. • The number range for a virtual interface instance varies, depending on the interface type. |
| Step 3 | service-policy type performance-traffic input <i>policy-name</i> Example: <pre>RP/0/RP0/CPU0:router(config-if)# service-policy type performance-traffic input policy1</pre> | Attaches the policy to the interface in the ingress direction. |
| Step 4 | commit | |

Configuration Examples for Implementing Video Monitoring

Scenario-1

An ethernet bundle interface has three physical members over which multicast video traffic is flowing at 300 pps for each flow.

Use video monitoring to monitor all the flows on this ethernet bundle, and raise a critical-level alarm, if the per-flow traffic load is over 10 % of expected rate. Raise an error-level alarm if the delay factor is greater than 4 ms. Report the collected statistics every 10 seconds. As long as the flow is active, keep the reported statistics for 10 minutes. Remove flow statistics if no packets are received for 30 seconds.

Example

```
ipv4 access-list sample-acl
 10 permit udp any any
!
```

```

class-map type traffic match-any sample-class
match access-group ipv4 sample-acl
end-class-map
!
policy-map type performance-traffic sample-policy
class type traffic sample-class
monitor parameters
  interval duration 10
  history 60
  timeout 3
!
monitor metric ip-cbr
  rate layer3 packet 300 pps
!
react 100 mrv
  threshold type immediate
  threshold value gt 10.00
  action syslog
  alarm severity error
  alarm type discrete
!
react 101 delay-factor
  threshold type immediate
  threshold value gt 4.00
  action syslog
  alarm severity error
  alarm type discrete
!
end-policy-map
!
interface Bundle-Ether10
  ipv4 address 172.192.1.1 255.255.255.0
  service-policy type performance-traffic input sample-policy
!
interface TenGigE0/6/0/0
  bundle id 10 mode on
!
interface TenGigE0/6/0/1
  bundle id 10 mode on
!
interface TenGigE0/6/0/2
  bundle id 10 mode on
!

```

Scenario-2

A VLAN subinterface is carrying 100 video streams with a common multicast group address of 225.0.0.1 and varying UDP port numbers. The expected packet rate at IP layer is unknown, but the media bit rate is known to be 1052800 bps. The media payload is known to contain MPEG-2 encoded CBR flows and default packetization is used (that is, in one UDP payload, there are seven MPEG packets, where each packet is 188 bytes long).

Do not monitor over 100 flows. Do not timeout and delete any flow even if flow stops, but raise an error-level alarm if the percentage of the stopped flows is over 90 %.

Example

```

ipv4 access-list sample-acl
  10 permit udp any host 225.0.0.1

```

```

!
class-map type traffic match-any sample-class
  match access-group ipv4 sample-acl
end-class-map
!
policy-map type performance-traffic sample-policy
  class type traffic sample-class
  monitor parameters
    flows 100
!
  monitor metric ip-cbr
    rate media 1052800 bps
  !
  react 100 media-stop
  action syslog
  alarm severity error
  alarm type grouped percent 90
  !
end-policy-map
!
interface GigabitEthernet0/0/0/0
  no shutdown
!
interface GigabitEthernet0/0/0/0.1
  encapsulation dot1q 500
  ipv4 address 172.192.1.1 255.255.255.0
  service-policy type performance-traffic input sample-policy
!

```

Under **monitor metric ip-cbr**, these two lines need not be configured as they are defaults:

- media packet count in-layer3 7
- media packet size 188

However, if these parameters are different from default values, they need to be configured.

Scenario-3

A main interface has three groups of multicast streams where the first group has UDP destination port of 1000, the second group has 2000, and the third group has 3000 and 4000. These three groups of streams flow at 100 pps, 200 pps, and 300 pps respectively.

Limit the maximum number of flows in each group to 300 flows and raise the error-level alarm, when they reach 90 % of the provisioned flow capacity.

Example

```

ipv4 access-list sample-acl-1
  10 permit udp any any eq 1000
!
ipv4 access-list sample-acl-2
  10 permit udp any any eq 2000
!
ipv4 access-list sample-acl-3
  10 permit udp any any eq 3000
  20 permit udp any any eq 4000
!
class-map type traffic match-any sample-class-1
  match access-group ipv4 sample-acl-1

```

```

    end-class-map
  !
  class-map type traffic match-any sample-class-2
  match access-group ipv4 sample-acl-2
  end-class-map
  !
  class-map type traffic match-any sample-class-3
  match access-group ipv4 sample-acl-3
  end-class-map
  !
  policy-map type performance-traffic sample-policy
  class type traffic sample-class-1
  monitor parameters
    interval duration 10
    history 60
    timeout 3
    flows 300
  !
  monitor metric ip-cbr
    rate layer3 packet 100 pps
  !
  react 100 flow-count
  threshold type immediate
  threshold value gt 270
  action syslog
  alarm severity error
  !
  class type traffic sample-class-2
  monitor parameters
    interval duration 10
    history 60
    timeout 3
    flows 300
  !
  monitor metric ip-cbr
    rate layer3 packet 200 pps
  !
  react 100 flow-count
  threshold type immediate
  threshold value gt 270
  action syslog
  alarm severity error
  !
  class type traffic sample-class-1
  monitor parameters
    interval duration 10
    history 60
    timeout 3
    flows 300
  !
  monitor metric ip-cbr
    rate layer3 packet 300 pps
  !
  react 100 flow-count
  threshold type immediate
  threshold value gt 270
  action syslog
  alarm severity error
  !
  !
  end-policy-map
  !
  interface GigabitEthernet0/0/0/0
  ipv4 address 172.192.1.1 255.255.255.0

```



```

service-policy type performance-traffic input sample-policy
!

```

Scenario-4

A 10GE main interface receives six high definition (HD) video streams from the digital contents manager (DCM), directly connected to six HD cameras in a sports stadium. Each HD video stream is uncompressed and its bandwidth is as high as 1.611 Gbps at layer 2, which is equivalent to 140625 pps. These six streams are received with multicast groups of 225.0.0.1 through 225.0.0.6, and the UDP port number is 5000.

Raise a critical-level alarm when the delay factor of any flow is above 2 ms, or media loss ratio is above 5 %. Use 10s interval and keep maximum history. Do not monitor more than 6 flows on this interface. Do not time out inactive flows.

Example

```

ipv4 access-list sample-acl
 10 permit udp any eq 5000 225.0.0.0/24 eq 5000
!
class-map type traffic match-any sample-class
 match access-group ipv4 sample-acl
end-class-map
!
policy-map type performance-traffic sample-policy
 class type traffic sample-class
  monitor parameters
   interval duration 10
   history 60
   flows 6
  !
  monitor metric ip-cbr
   rate layer3 packet 140625 pps
  !
  react 100 mrv
   threshold type immediate
   threshold value gt 5.00
   action syslog
   alarm severity critical
   alarm type discrete
  !
  react 200 delay-factor
   threshold type immediate
   threshold value gt 2.00
   action syslog
   alarm severity critical
   alarm type discrete
  !
end-policy-map
!
interface TenGigE0/2/0/0
 ipv4 address 172.192.1.1 255.255.255.0
 service-policy type performance-traffic input sample-policy
!

```

Additional References

Related Documents

| Related Topic | Document Title |
|---|--|
| Multicast command reference document | <i>Multicast Command Reference for Cisco CRS Routers</i> |
| Getting started material | <i>Cisco IOS XR Getting Started Guide for the Cisco CRS Router</i> |
| Modular quality of service command reference document | <i>Modular QoS Command Reference for Cisco CRS Routers</i> |

MIBs

| MIBs | MIBs Link |
|------|--|
| — | To locate and download MIBs using Cisco IOS XR software, use the Cisco MIB Locator found at the following URL and choose a platform under the Cisco Access Products menu: https://mibs.cloudapps.cisco.com/ITDIT/MIBS/servlet/index |

RFCs

| RFCs | Title |
|---------|-------------------------------------|
| RFC4445 | Proposed Media Delivery Index (MDI) |

Technical Assistance

| Description | Link |
|---|---|
| The Cisco Technical Support website contains thousands of pages of searchable technical content, including links to products, technologies, solutions, technical tips, and tools. Registered Cisco.com users can log in from this page to access even more content. | http://www.cisco.com/techsupport |



CHAPTER 8

Implementing Cisco Express Forwarding

Cisco Express Forwarding (CEF) is advanced, Layer 3 IP switching technology. CEF optimizes network performance and scalability for networks with large and dynamic traffic patterns, such as the Internet, on networks characterized by intensive web-based applications, or interactive sessions.



Note For complete descriptions of the CEF commands listed in this module, you can refer to the [Related Documents, on page 173](#) section of this module.

Feature History for Implementing CEF

| Release | Modification |
|---------------|---|
| Release 2.0 | This feature was introduced. |
| Release 3.3.0 | Loose and Strict support for uRPF was added. The CEF Nonrecursive Accounting feature was removed. |
| Release 3.6.0 | Per-flow load balancing feature was added. |
| Release 3.7.0 | The OSPFv2 SPF Prefix Prioritization feature was added. The show cef bgp-attribute command was added. |

- [Prerequisites for Implementing Cisco Express Forwarding, on page 139](#)
- [Information About Implementing Cisco Express Forwarding Software, on page 140](#)
- [How to Implement CEF, on page 145](#)
- [Configuration Examples for Implementing CEF on Routers Software, on page 153](#)
- [Additional References, on page 173](#)

Prerequisites for Implementing Cisco Express Forwarding

The following prerequisites are required to implement Cisco Express Forwarding:

- You must be in a user group associated with a task group that includes the proper task IDs. The command reference guides include the task IDs required for each command. If you suspect user group assignment is preventing you from using a command, contact your AAA administrator for assistance.

Information About Implementing Cisco Express Forwarding Software

To implement Cisco Express Forwarding features in this document you must understand the following concepts:

Key Features Supported in the Cisco Express Forwarding Implementation

The following features are supported for CEF on Cisco IOS XR software:

- Border Gateway Protocol (BGP) policy accounting
- Reverse path forwarding (RPF)
- Virtual interface support
- Multipath support
- Route consistency
- High availability features such as packaging, restartability, and Out of Resource (OOR) handling
- OSPFv2 SPF prefix prioritization
- BGP attributes download

Benefits of CEF

CEF offers the following benefits:

- Improved performance—CEF is less CPU-intensive than fast-switching route caching. More CPU processing power can be dedicated to Layer 3 services such as quality of service (QoS) and encryption.
- Scalability—CEF offers full switching capacity at each line card.
- Resilience—CEF offers an unprecedented level of switching consistency and stability in large dynamic networks. In dynamic networks, fast-switched cache entries are frequently invalidated due to routing changes. These changes can cause traffic to be process switched using the routing table, rather than fast switched using the route cache. Because the Forwarding Information Base (FIB) lookup table contains all known routes that exist in the routing table, it eliminates route cache maintenance and the fast-switch or process-switch forwarding scenario. CEF can switch traffic more efficiently than typical demand caching schemes.

CEF Components

Cisco IOS XR software CEF always operates in CEF mode with two distinct components: a Forwarding Information Base (FIB) database and adjacency table—a protocol-independent adjacency information base (AIB).

CEF is a primary IP packet-forwarding database for Cisco IOS XR software. CEF is responsible for the following functions:

- Software switching path
- Maintaining forwarding table and adjacency tables (which are maintained by the AIB) for software and hardware forwarding engines

The following CEF forwarding tables are maintained in Cisco IOS XR software:

- IPv4 CEF database
- IPv6 CEF database
- MPLS LFD database
- Multicast Forwarding Table (MFD)

The protocol-dependent FIB process maintains the forwarding tables for IPv4 and IPv6 unicast in the route processor (RP) and each MSC.

The FIB on each node processes Routing Information Base (RIB) updates, performing route resolution and maintaining FIB tables independently in the RP and each MSC. FIB tables on each node can be slightly different. Adjacency FIB entries are maintained only on a local node, and adjacency entries linked to FIB entries could be different.

Border Gateway Protocol Policy Accounting

Border Gateway Protocol (BGP) policy accounting measures and classifies IP traffic that is sent to, or received from, different peers. Policy accounting is enabled on an individual input or output interface basis, and counters based on parameters such as community list, autonomous system number, or autonomous system path are assigned to identify the IP traffic.



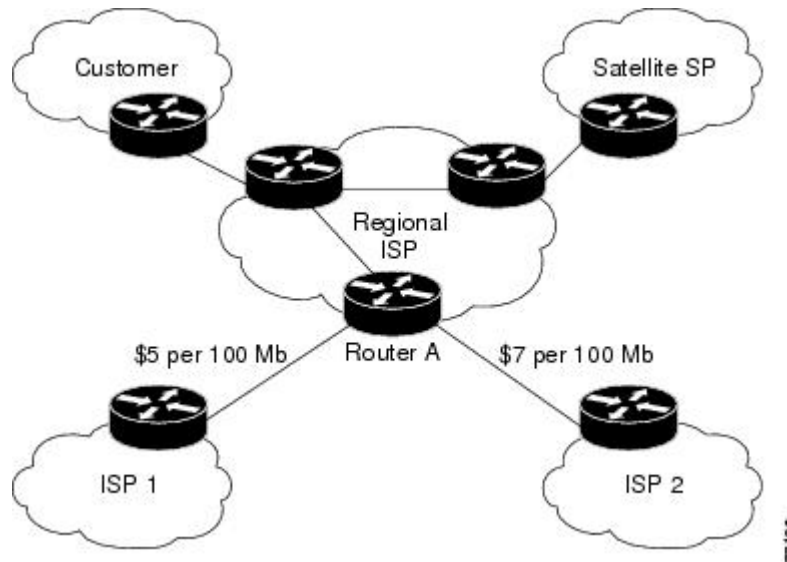
Note There are two types of route policies. The first type (regular BGP route policies) is used to filter the BGP routes advertised into or out from the BGP links. This type of route policy is applied to the specific BGP neighbor. The second type (specific route policy) is used to set up a traffic index for the BGP prefixes. This route policy is applied to the global BGP IPv4 address family to set up the traffic index when the BGP routes are inserted into the RIB table. BGP policy accounting uses the second type of route policy.

Using BGP policy accounting, you can account for traffic according to the route it traverses. Service providers can identify and account for all traffic by customer and bill accordingly. In [Figure 13: Sample Topology for BGP Policy Accounting, on page 142](#), BGP policy accounting can be implemented in Router A to measure packet and byte volumes in autonomous system buckets. Customers are billed appropriately for traffic that is routed from a domestic, international, or satellite source.



Note BGP policy accounting measures and classifies IP traffic for BGP prefixes only.

Figure 13: Sample Topology for BGP Policy Accounting



Based on the specified routing policy, BGP policy accounting assigns each prefix a traffic index (bucket) associated with an interface. BGP prefixes are downloaded from the Routing Information Base (RIB) to the FIB along with the traffic index.

There are a total of 63 (1 to 63) traffic indexes (bucket numbers) that can be assigned for BGP prefixes. Internally, there is an accounting table associated with the traffic indexes to be created for each input (ingress) and output (egress) interface. The traffic indexes allow you to account for the IP traffic, where the source IP address, the destination IP address, or both are BGP prefixes.



Note Traffic index 0 contains the packet count using Interior Gateway Protocol (IGP) routes.

Reverse Path Forwarding (Strict and Loose)

Unicast IPv4 and IPv6 Reverse Path Forwarding (uRPF), both strict and loose modes, help mitigate problems caused by the introduction of malformed or spoofed IP source addresses into a network by discarding IP packets that lack a verifiable IP source address. Unicast RPF does this by doing a reverse lookup in the CEF table. Therefore, Unicast Reverse Path Forwarding is possible only if CEF is enabled on the router.



Note Unicast RPF allows packets with 0.0.0.0 source addresses and 255.255.255.255 destination addresses to pass so that Bootstrap Protocol and Dynamic Host Configuration Protocol (DHCP) will function properly.

When strict uRPF is enabled, the source address of the packet is checked in the FIB. If the packet is received on the same interface that would be used to forward the traffic to the source of the packet, the packet passes the check and is further processed; otherwise, it is dropped. Strict uRPF should only be applied where there is natural or configured symmetry. Because internal interfaces are likely to have routing asymmetry, that is,

multiple routes to the source of a packet, strict uRPF should not be implemented on interfaces that are internal to the network.



Note The behavior of strict RPF varies slightly by platform, number of recursion levels, and number of paths in Equal-Cost Multipath (ECMP) scenarios. A platform may switch to loose RPF check for some or all prefixes, even though strict RPF is configured.

When loose uRPF is enabled, the source address of the packet is checked in the FIB. If it exists and matches a valid forwarding entry, the packet passes the check and is further processed; otherwise, it is dropped.

Loose and strict uRPF supports two options: **allow self-ping** and **allow default**. The **self-ping** option allows the source of the packet to ping itself. The **allow default** option allows the lookup result to match a default routing entry. When the **allow default** option is enabled with the strict mode of the uRPF, the packet is processed further only if it arrived through the default interface.

Route Processor Management Ethernet Forwarding

Forwarding from the MSC interface to the RP Management Ethernet is disabled by default. The **rp mgmtethernet forwarding** command is used to enable forwarding from the MSC interface to RP Management Ethernet.

Forwarding from the RP Management Ethernet to the MSC interface, and from the RP Management Ethernet to RP Management Ethernet, is enabled by default.

Per-Flow Load Balancing

Load balancing describes the functionality in a router that distributes packets across multiple links based on Layer 3 (network layer) and Layer 4 (transport layer) routing information. If the router discovers multiple paths to a destination, the routing table is updated with multiple entries for that destination.

Per-flow load balancing performs these functions:

- Incoming data traffic is evenly distributed over multiple equal-cost connections within a bundle interface.
- Layer 2 bundle and Layer 3 (network layer) load balancing decisions are taken on IPv4, IPv6, and MPLS flows, which are supported for the 7-tuple hash algorithm.
- A 7-tuple hash algorithm provides more granular load balancing than the existing 3-tuple hash algorithm.
- The same hash algorithm (3-tuple or 7-tuple) is used for load balancing over multiple equal-cost Layer 3 (network layer) paths. The Layer 3 (network layer) path is on a physical interface or on a bundle interface. In addition, load balancing over member links can occur within a Layer 2 bundle interface.
- The **cef load-balancing fields** command allows you to select either the 3-tuple hash algorithm (default) or the 7-tuple hash algorithm.

Layer 3 (Network Layer) Routing Information

The 3-tuple load-balance hash calculation contains these Layer 3 (Network Layer) inputs:

- Source IP address

- Destination IP address
- Router ID

The 7-tuple load-balance hash calculation contains 3-tuple inputs and these additional following Layer 4 (Transport Layer) inputs:

Layer 4 (Transport Layer) Routing Information

The 5-tuple load-balance hash calculation contains 3-tuple inputs and these additional following Layer 4 (Transport Layer) inputs:

- Source port
- Destination port
- Protocol
- Ingress interface handle



Note For MPLS packets with non-IPv4/non-IPv6 payload, when CRS is used as a core router, these fields are considered for load balancing:

- 3 tuple hashing - MPLS Label, Router ID
- 7 tuple hashing - MPLS Label, Router ID, Ingress interface handle

For more information about configuring the hashing algorithm (**cef load-balancing fields l4** command) that is used for load balancing during forwarding, see the *IP Addresses and Services Command Reference for Cisco CRS Routers*.



Note In load-balancing scenarios, a line card may not use all output paths downloaded from routing protocols. This behavior varies with platform, number of recursion levels, and the fact whether MPLS is involved, or not.

BGP Attributes Download

The BGP Attributes Download feature enables you to display the installed BGP attributes in CEF. Configure the **show cef bgp-attribute** command to display the installed BGP attributes in CEF. You can use the **show cef bgp-attribute attribute-id** command and the **show cef bgp-attribute local-attribute-id** command to look at specific BGP attributes by attribute ID and local attribute ID.

Verification

```
Router# show cef bgp-attribute
Wed Aug 21 14:05:51.772 UTC
```

```
VRF: default
```

```
Table ID: 0xe0000000. Total number of entries: 1
```



```
OOB state: GREEN. Number of OOB attributes: 0

BGP Attribute ID: 0x6, Local Attribute ID: 0x1
  Aspath      : 2
  Community   :
  Origin AS   : 2
  Next Hop AS : 2
```

GTP Tunnel Load Balancing

GPRS Tunneling Protocol (GTP) is used mainly to deliver mobile data on wireless networks via Cisco CRS Router as core router. When two routers carrying GTP traffic are connected with link bundling, the traffic is required to be distributed evenly between all bundle members. You can use the **bundle-hash** command in EXEC mode to verify that the interface selected within the bundle for load balancing matches with the output from the **bundle-hash** command.

When two routers carrying GTP traffic are connected with equal-cost multi-path (ECMP) between them, you can use the **sh cef exact-route** command in EXEC mode to verify the interface selected for load balancing.

To achieve load balancing, Cisco CRS router uses 7-tuple load balancing mechanism which takes account of source IP, destination IP, router-id, ingress interface, protocol, L4 source and destination port (if traffic is TCP or UDP) fields from the packet. But for GTP traffic, limited number of unique values for these fields restrict the equal distribution of traffic load on tunnel.

In order to avoid the polarization for GTP traffic in load balancing, a tunnel endpoint identifier (TEID) in GTP header is used instead of UDP port number. Since TEID is unique per tunnel, traffic can be evenly load balanced across multiple links in the bundle.

GTP tunnel load balancing feature adds support for:

- GTP with IPv4/IPv6 transport header on physical interface
- GTP traffic over TE tunnel
- GTPv1-U with UDP port 2152

The **cef load-balancing fields L4** command enables the GTP tunnel load balancing.

To know the egress interface for GTP traffic after load balancing, use **show cef {ipv4 | ipv6} exact-route** command with TEID in place of L4 protocol source and destination port number. Use 16MSBbits of TEID in source port and 16LSBbits of TEID in destination port.

How to Implement CEF

This section contains instructions for the following tasks:

Verifying CEF

This task allows you to verify CEF.

SUMMARY STEPS

1. **show cef {ipv4 | ipv6}**
2. **show cef {ipv4 | ipv6} summary**

3. `show cef {ipv4 | ipv6} detail`
4. `show adjacency detail`

DETAILED STEPS

| | Command or Action | Purpose |
|---------------|---|--|
| Step 1 | show cef {ipv4 ipv6} Example: RP/0/RP0/CPU0:router# show cef ipv4 | Displays the IPv4 or IPv6 CEF table. The next hop and forwarding interface are displayed for each prefix. Note The output of the show cef command varies by location. |
| Step 2 | show cef {ipv4 ipv6} summary Example: RP/0/RP0/CPU0:router# show cef ipv4 summary | Displays a summary of the IPv4 or IPv6 CEF table. |
| Step 3 | show cef {ipv4 ipv6} detail Example: RP/0/RP0/CPU0:router# show cef ipv4 detail | Displays detailed IPv4 or IPv6 CEF table information. |
| Step 4 | <code>show adjacency detail</code> Example: RP/0/RP0/CPU0:router# show adjacency detail | Displays detailed adjacency information, including Layer 2 information for each interface. Note The output of the show adjacency command varies by location. |

Configuring BGP Policy Accounting

This task allows you to configure BGP policy accounting.



Note There are two types of route policies. BGP policy accounting uses the type that is used to set up a traffic index for the BGP prefixes. The route policy is applied to the global BGP IPv4 address family to set up the traffic index when the BGP routes are inserted into the RIB table.

BGP policy accounting enables per interface accounting for ingress and egress IP traffic based on the traffic index assigned to the source IP address (BGP prefix) and destination IP address (BGP prefix). The traffic index of BGP prefixes can be assigned according to the following parameters using Routing Policy Language (RPL):

- prefix-set
- AS-path-set
- community-set



Note BGP policy accounting is supported on IPv4 prefixes only.

Two configuration tasks provide the ability to classify BGP prefixes that are in the RIB according to the prefix-set, AS-path-set, or the community-set parameters:

1. Use the **route-policy** command to define the policy for traffic index setup based on the prefix-set, AS-path-set, or community-set.
2. Use the BGP **table-policy** command to apply the defined route policy to the global BGP IPv4 unicast address family.

See the *Routing Command Reference for Cisco CRS Routers* for information on the **route-policy** and **table-policy** commands.

BGP policy accounting can be enabled on each interface with the following options:

- Use the `ipv4 bgp policy accounting` command with one of the following keyword options:
 - `input source-accounting`
 - `input destination-accounting`
 - `input source-accounting destination-accounting`
- Use the `ipv4 bgp policy accounting` command with one of the following keyword options:
 - `output source-accounting`
 - `output destination-accounting`
 - `output source-accounting destination-accounting`
- Use any combination of the keywords provided for the **ipv4 bgp policy accounting** command.

Before you begin

Before using the BGP policy accounting feature, you must enable BGP on the router (CEF is enabled by default). See the *Routing Configuration Guide for Cisco CRS Routers* for information on enabling BGP.

Verifying BGP Policy Accounting

This task allows you to verify BGP policy accounting.



Note BGP policy accounting is supported on IPv4 prefixes.

Before you begin

BGP policy accounting must be configured. See the [Configuring BGP Policy Accounting, on page 146](#).

SUMMARY STEPS

1. **show route bgp**
2. **show bgp summary**
3. **show bgp ip-address**
4. **show route ipv4 ip-address**
5. **show cef ipv4 prefix**
6. **show cef ipv4 prefix detail**
7. **show cef ipv4 interface type interface-path-id bgp-policy-statistics**

DETAILED STEPS

| | Command or Action | Purpose |
|---------------|--|---|
| Step 1 | show route bgp Example: RP/0/RP0/CPU0:router# show route bgp | Displays all BGP routes with traffic indexes. |
| Step 2 | show bgp summary Example: RP/0/RP0/CPU0:router# show bgp summary | Displays the status of all BGP neighbors. |
| Step 3 | show bgp ip-address Example: RP/0/RP0/CPU0:router# show bgp 40.1.1.1 | Displays BGP prefixes with BGP attributes. |
| Step 4 | show route ipv4 ip-address Example: RP/0/RP0/CPU0:router# show route ipv4 40.1.1.1 | Displays the specific BGP route with the traffic index in the RIB. |
| Step 5 | show cef ipv4 prefix Example: RP/0/RP0/CPU0:router# show cef ipv4 40.1.1.1 | Displays the specific BGP prefix with the traffic index in the RP FIB. |
| Step 6 | show cef ipv4 prefix detail Example: RP/0/RP0/CPU0:router# show cef ipv4 40.1.1.1 detail | Displays the specific BGP prefix with detailed information in the RP FIB. |
| Step 7 | show cef ipv4 interface type interface-path-id bgp-policy-statistics Example: | Displays the BGP Policy Accounting statistics for the specific interface. |

| | Command or Action | Purpose |
|--|---|---------|
| | RP/0/RP0/CPU0:router# show cef ipv4 interface TenGigE 0/2/0/4 bgp-policy-statistics | |

Configuring a Route Purge Delay

This task allows you to configure a route purge delay. A purge delay purges routes when the RIB or other related process experiences a failure.

SUMMARY STEPS

1. **configure**
2. **cef purge-delay** *seconds*
3. **commit**

DETAILED STEPS

| | Command or Action | Purpose |
|---------------|---|---|
| Step 1 | configure | |
| Step 2 | cef purge-delay <i>seconds</i> Example: RP/0/RP0/CPU0:router(config)# cef purge-delay 180 | Configures a delay in purging routes when the Routing Information Base (RIB) or other related processes experience a failure. |
| Step 3 | commit | |

Configuring Unicast RPF Checking

This task allows you to configure unicast Reverse Path Forwarding (uRPF) checking. Unicast RPF checking allows you to mitigate problems caused by malformed or forged (spoofed) IP source addresses that pass through a router. Malformed or forged source addresses can indicate denial-of-service (DoS) attacks based on source IP address spoofing.

SUMMARY STEPS

1. **configure**
2. **interface** *type interface-path-id*
3. **{ipv4 | ipv6} verify unicast source reachable-via** {any | rx} [allow-default] [allow-self-ping]
4. **commit**

DETAILED STEPS

| | Command or Action | Purpose |
|---------------|-------------------|---------|
| Step 1 | configure | |

| | Command or Action | Purpose |
|---------------|---|---|
| Step 2 | interface <i>type interface-path-id</i> Example: <pre>RP/0/RP0/CPU0:router(config)# interface GigabitEthernet 0/1/0/0</pre> | Enters interface configuration mode. |
| Step 3 | {ipv4 ipv6} verify unicast source reachable-via {any rx} [allow-default] [allow-self-ping] Example: <pre>RP/0/RP0/CPU0:router(config-if)# ipv4 verify unicast source reachable-via rx</pre> | Enables IPv4 or IPv6 uRPF checking. <ul style="list-style-type: none"> • The rx keyword enables strict unicast RPF checking. If strict unicast RPF is enabled, a packet is not forwarded unless its source prefix exists in the routing table and the output interface matches the interface on which the packet was received. • The allow-default keyword enables the matching of default routes. This option applies to both loose and strict RPF. • The allow-self-ping keyword enables the router to ping out an interface. This option applies to both loose and strict RPF. <p>Note IPv6 uRPF checking is not supported on ASR 9000 Ethernet line cards.</p> |
| Step 4 | commit | |

Configuring Modular Services Card-to-Route Processor Management Ethernet Interface Switching

This task allows you to enable MSC-to-RP management Ethernet interface switching.

SUMMARY STEPS

1. **configure**
2. **rp mgmtethernet forwarding**
3. **commit**

DETAILED STEPS

| | Command or Action | Purpose |
|---------------|---|---|
| Step 1 | configure | |
| Step 2 | rp mgmtethernet forwarding Example: <pre>RP/0/RP0/CPU0:router(config)# rp mgmtethernet forwarding</pre> | Enables switching from the MSC to the route processor Management Ethernet interfaces. |

| | Command or Action | Purpose |
|--------|-------------------|---------|
| Step 3 | commit | |

Configuring Per-Flow Load Balancing

This section describes the following tasks to configure per-flow load balancing:

Configuring a 7-Tuple Hash Algorithm

This task allows you to configure per-flow load balancing for a 7-tuple hash algorithm.

SUMMARY STEPS

1. **configure**
2. **cef load-balancing fields {L3 | L4}**
3. **commit**
4. **show cef {ipv4 | ipv6} summary [location node-id]**

DETAILED STEPS

| | Command or Action | Purpose |
|--------|--|---|
| Step 1 | configure | |
| Step 2 | cef load-balancing fields {L3 L4} Example: <pre>RP/0/RP0/CPU0:router(config)# cef load-balancing fields L4</pre> | <p>Configures the hashing algorithm that is used for load balancing during forwarding. The example shows that the L4 field is selected.</p> <ul style="list-style-type: none"> • Use the L3 keyword to specify the Layer 3 load-balancing for the hash algorithm <p>Since L3 is configured as the default value, you do not need to use the cef load-balancing fields command unless you want to configure Layer 4.</p> <ul style="list-style-type: none"> • Use the L4 keyword to specify the Layer 3 and Layer 4 load-balancing for the hash algorithm. <p>For a list of the inputs for Layer 3 and Layer 4, see Per-Flow Load Balancing, on page 143.</p> |
| Step 3 | commit | |
| Step 4 | show cef {ipv4 ipv6} summary [location node-id] Example: <pre>RP/0/RP0/CPU0:router# show cef ipv4 summary</pre> | <p>Displays the load balancing field for the IPv4 or IPv6 CEF table.</p> <ul style="list-style-type: none"> • (Optional) Use the location keyword display a summary of the IPv4 CEF table for the designated node. The <i>node-id</i> argument is entered in the <i>rack/slot/module</i> notation |

Verifying the CEF Exact Route with 7-Tuple Parameters

The following 7-tuple parameters are specified to obtain the CEF exact route for both IPv4 and IPv6:

- Source address
- Destination address
- Source port and range of destination ports
- Protocol
- Ingress interface
- Router ID

To display the path an MPLS flow would take, use the **show mpls forwarding exact-route** command. The MPLS flow comprises a source address and a destination address.

To display the path a bundle flow would take, use the **bundle-hash** command. The bundle flow comprises a source and a destination address. For more information, see *Interface and Hardware Component Command Reference for Cisco CRS Routers*.

To verify the IPv4 7-tuple parameters, perform the following steps:

SUMMARY STEPS

1. Configure parallel interfaces between back-to-back routers.
2. Create route traffic streams so that there is a stream placed onto each configured interface.
3. Use the **show cef ipv4 exact-route** command in EXEC mode to verify that the interface selected for load balancing matches with the output from this command. The following example shows the exact route for the Layer 4 information:
4. Configure Equal Cost Multipath Protocol (ECMP) interfaces, for example, between back-to-back routers.
5. Create route traffic streams so that there is a stream placed onto each configured interface.
6. Use the **show cef ipv6 exact-route** command in EXEC mode to verify that the interface selected for load balancing matches with the output from this command. The following example shows the exact route for the Layer 4 information:

DETAILED STEPS

-
- Step 1** Configure parallel interfaces between back-to-back routers.
- Step 2** Create route traffic streams so that there is a stream placed onto each configured interface.
- Step 3** Use the **show cef ipv4 exact-route** command in EXEC mode to verify that the interface selected for load balancing matches with the output from this command. The following example shows the exact route for the Layer 4 information:

Example:

```
RP/0/RP0/CPU0:router# show cef ipv4 exact-route 20 .6.1.9 22.6.1.9 protocol udp source-port 1
destination-port 1 ingress-interface GigabitEthernet 0/1/0/4
```

```
22.6.1.9/32 version 0, internal 0x40040001 (0x78439fd0) [3], 0x0 (0x78aaf928), 0x4400 (0x78ed62d0)
remote adjacency to GigabitEthernet0/1/4/4 Prefix Len 32, traffic index 0, precedence routine (0)
via GigabitEthernet0/1/4/4
```


To verify the IPv6 7-tuple parameters, perform the following steps:

- Step 4** Configure Equal Cost Multipath Protocol (ECMP) interfaces, for example, between back-to-back routers.
- Step 5** Create route traffic streams so that there is a stream placed onto each configured interface.
- Step 6** Use the **show cef ipv6 exact-route** command in EXEC mode to verify that the interface selected for load balancing matches with the output from this command. The following example shows the exact route for the Layer 4 information:

Example:

```
RP/0/RP0/CPU0:router# show cef ipv6 exact-route 20:6:1::9 22:6:1::9 protocol udp source-port 1
destination-port 1 ingress-interface GigabitEthernet 0/1/0/4
```

```
22:6:1::/64, version 0, internal 0x40000001 (0x7846c048) [3], 0x0 (0x78aea3d0), 0x0 (0x0) remote
adjacency to GigabitEthernet0/1/4/4 Prefix Len 64, traffic index 0, precedence routine (0)
via GigabitEthernet0/1/4/4
```

Configuring BGP Attributes Download

This task allows you to configure the BGP Attributes Download feature.

Configuring BGP Attributes Download

SUMMARY STEPS

1. **configure**
2. **cef bgp attribute** {*attribute-id* | *local-attribute-id* }
3. **commit**

DETAILED STEPS

| | Command or Action | Purpose |
|---------------|---|---------------------------------|
| Step 1 | configure | |
| Step 2 | cef bgp attribute { <i>attribute-id</i> <i>local-attribute-id</i> } Example: RP/0/RP0/CPU0:router(config)# cef bgp attribute { <i>attribute-id</i> <i>local-attribute-id</i> } | Configures a CEF BGP attribute. |
| Step 3 | commit | |

Configuration Examples for Implementing CEF on Routers Software

This section provides the following configuration examples:

Configuring BGP Policy Accounting: Example

The following example shows how to configure BGP policy accounting.

Configure loopback interfaces for BGP router-id:

```
interface Loopback1
  ipv4 address
  190.1.1.1 255.255.255.255
```

Configure interfaces with the BGP policy accounting options:

```
interface TenGigE0/2/0/2
  mtu 1514
  ipv4 address
  17.1.0.1 255.255.255.0
  proxy-arp
  ipv4 directed-broadcast
  ipv4 bgp policy accounting input source-accounting destination-accounting
  ipv4 bgp policy accounting output source-accounting destination-accounting
  !
interface TenGigE0/2/0/2.1
  ipv4 address
  17.1.1.1 255.255.255.0
  ipv4 bgp policy accounting input source-accounting destination-accounting
  ipv4 bgp policy accounting output source-accounting destination-accounting
  encapsulation dot1q 1
  !
interface TenGigE0/2/0/4
  mtu 1514
  ipv4 address
  18.1.0.1 255.255.255.0
  proxy-arp
  ipv4 directed-broadcast
  ipv4 bgp policy accounting input source-accounting destination-accounting
  ipv4 bgp policy accounting output source-accounting destination-accounting
  !
interface TenGigE0/2/0/4.1
  ipv4 address
  18.1.
  1.1 255.255.255.0
  ipv4 bgp policy accounting input source-accounting destination-accounting
  ipv4 bgp policy accounting output source-accounting destination-accounting
  encapsulation dot1q 1
  !
interface GigabitEthernet 0/0/0/4
  mtu 4474
  ipv4 address
  4.1.0.
  1 255.255.0.0
  ipv4 directed-broadcast
  ipv4 bgp policy accounting input source-accounting destination-accounting
  ipv4 bgp policy accounting output source-accounting destination-accounting
  encapsulation ppp
  GigabitEthernet
  crc 32
  !
  keepalive disable
  !
interface GigabitEthernet 0/0/0/8
  mtu 4474
  ipv4 address
```

```
8.
1.0.1 255.255.0.0
  ipv4 directed-broadcast
  ipv4 bgp policy accounting input source-accounting destination-accounting
  ipv4 bgp policy accounting output source-accounting destination-accounting
  GigabitEthernet
    crc 32
  !
  keepalive disable
!
```

Configure controller:

```
controller GigabitEthernet 0/0/0/4
  ais-shut
  path
    ais-shut
  !
  threshold sf-ber 5
!
controller SONET0/0/0/8
  ais-shut
  path
    ais-shut
  !
  threshold sf-ber 5
!
```

Configure AS-path-set and prefix-set:

```
as-path-set as107
  ios-regex '107$'
end-set

as-path-set as108
  ios-regex '108$'
end-set

prefix-set RT-65.0
  65.0.0.0/16 ge 16 le 32
end-set

prefix-set RT-66.0
  66.0.0.0/16 ge 16 le 32
end-set
```

Configure the route-policy (table-policy) to set up the traffic indexes based on each prefix, AS-path-set, and prefix-set:

```
route-policy bpa1

  if destination in (
27.1.1.0/24) then
    set traffic-index 1
  elseif destination in (
27.1.2.0/24) then
    set traffic-index 2
  elseif destination in (
27.1.3.0/24) then
    set traffic-index 3
  elseif destination in (
27.1.4.0/24) then
    set traffic-index 4
```

```

    elseif destination in (
27.1.5.0/24) then
        set traffic-index 5
    endif

    if destination in (
28.1.1.0/24) then
        set traffic-index 6
    elseif destination in (
28.1.2.0/24) then
        set traffic-index 7
    elseif destination in (
28.1.3.0/24) then
        set traffic-index 8
    elseif destination in (
28.1.4.0/24) then
        set traffic-index 9
    elseif destination in (
28.1.5.0/24) then
        set traffic-index 10
    endif

    if as-path in as107 then
        set traffic-index 7
    elseif as-path in as108 then
        set traffic-index 8
    endif

    if destination in RT-65.0 then
        set traffic-index 15
    elseif destination in RT-66.0 then
        set traffic-index 16
    endif

end-policy

```

Configure the regular BGP route-policy to pass or drop all the BGP routes:

```

route-policy drop-all
    drop
end-policy
!
route-policy pass-all
    pass
end-policy
!

```

Configure the BGP router and apply the table-policy to the global ipv4 address family:

```

router bgp 100
    bgp router-id Loopback1
    bgp graceful-restart
    bgp as-path-loopcheck
    address-family ipv4 unicast
        table-policy bpa1
        maximum-paths 8
    bgp dampening
!

```

Configure the BGP neighbor-group:

```

neighbor-group ebgp-peer-using-int-addr
    address-family ipv4 unicast

```

```

    policy pass-all in
    policy drop-all out
    !
    !
neighbor-group ebgp-peer-using-int-addr-121
  remote-as 121
  address-family ipv4 unicast
    policy pass-all in
    policy drop-all out
  !
  !
neighbor-group ebgp-peer-using-int-addr-pass-out
  address-family ipv4 unicast
    policy pass-all in
    policy pass-all out
  !
  !

```

Configure BGP neighbors:

```

neighbor
4.
1.0.2
  remote-as 107
  use neighbor-group ebgp-peer-using-int-addr
  !
neighbor
8.
1.0.2
  remote-as 108
  use neighbor-group ebgp-peer-using-int-addr
  !
neighbor
17.
1.0.2
  use neighbor-group ebgp-peer-using-int-addr-121
  !
neighbor
17.1.
1.2
  use neighbor-group ebgp-peer-using-int-addr-121
  !
neighbor
18.
1.0.2
  remote-as 122
  use neighbor-group ebgp-peer-using-int-addr
  !
neighbor
18.
1.1.2
  remote-as 1221
  use neighbor-group ebgp-peer-using-int-addr
  !
end

```

Verifying BGP Policy Statistics: Example

The following example shows how to verify the traffic index setup for each BGP prefix and BGP Policy Accounting statistics on ingress and egress interfaces. The following traffic stream is configured for this example:

- Traffic comes in from GigabitEthernet 0/2/0/4 and goes out to 5 VLAN subinterfaces under GigabitEthernet 0/2/0/2
- Traffic comes in from GigabitEthernet 0/0/0/8 and goes out to GigabitEthernet 0/0/0/4

```
show cef ipv4 interface GigabitEthernet 0/0/0/8 bgp-policy-statistics
```

```
GigabitEthernet0/0/0/8 is up
Input BGP policy accounting on dst IP address enabled
  buckets      packets      bytes
  7             5001160    500116000
  15            10002320   1000232000
Input BGP policy accounting on src IP address enabled
  buckets      packets      bytes
  8             5001160    500116000
  16            10002320   1000232000
Output BGP policy accounting on dst IP address enabled
  buckets      packets      bytes
  0             15          790
Output BGP policy accounting on src IP address enabled
  buckets      packets      bytes
  0             15          790
```

```
show cef ipv4 interface GigabitEthernet 0/0/0/4 bgp-policy-statistics
```

```
GigabitEthernet0/0/0/4 is up
Input BGP policy accounting on dst IP address enabled
  buckets      packets      bytes
Input BGP policy accounting on src IP address enabled
  buckets      packets      bytes
Output BGP policy accounting on dst IP address enabled
  buckets      packets      bytes
  0             13          653
  7             5001160    500116000
  15            10002320   1000232000
Output BGP policy accounting on src IP address enabled
  buckets      packets      bytes
  0             13          653
  8             5001160    500116000
  16            10002320   1000232000
```

```
show cef ipv4 interface GigabitEthernet 0/2/0/4 bgp-policy-statistics
```

```
GigabitEthernet0/2/0/4 is up
Input BGP policy accounting on dst IP address enabled
  buckets      packets      bytes
  1             3297102    329710200
  2             3297102    329710200
  3             3297102    329710200
  4             3297101    329710100
  5             3297101    329710100
Input BGP policy accounting on src IP address enabled
  buckets      packets      bytes
  6             3297102    329710200
  7             3297102    329710200
  8             3297102    329710200
  9             3297101    329710100
  10            3297101    329710100
Output BGP policy accounting on dst IP address enabled
  buckets      packets      bytes
  0             15          733
Output BGP policy accounting on src IP address enabled
  buckets      packets      bytes
```

```

0                15                733

show cef ipv4 interface GigabitEthernet 0/2/0/2.1 bgp-policy-statistics

GigabitEthernet 0/2/0/2.1 is up
Input BGP policy accounting on dst IP address enabled
  buckets      packets      bytes
Input BGP policy accounting on src IP address enabled
  buckets      packets      bytes
Output BGP policy accounting on dst IP address enabled
  buckets      packets      bytes
0                15                752
1                3297102      329710200
2                3297102      329710200
3                3297102      329710200
4                3297101      329710100
5                3297101      329710100
Output BGP policy accounting on src IP address enabled
  buckets      packets      bytes
0                15                752
6                3297102      329710200
7                3297102      329710200
8                3297102      329710200
9                3297101      329710100
10               3297101      329710100

```

The following example show how to verify BGP routes and traffic indexes:

```

show route bgp

B
 27.1.1.0/24 [20/0] via
17.
1.1.2, 00:07:09
  Traffic Index 1

B  27.1.2.0/24 [20/0] via
17.
1.1.2, 00:07:09
  Traffic Index 2

B
27.1.3.0/24 [20/0] via
17.
1.1.2, 00:07:09
  Traffic Index 3

B
27.1.4.0/24 [20/0] via
17.
1.1.2, 00:07:09
  Traffic Index 4

B
27.1.5.0/24 [20/0] via
17.
1.1.2, 00:07:09
  Traffic Index 5

B
28.
1.1.0/24 [20/0] via
18.
1.1.2, 00:07:09
  Traffic Index 6

B
28.
1.2.0/24 [20/0] via

```

```
18.
1.1.2, 00:07:09
    Traffic Index 7
B
28.
1.3.0/24 [20/0] via
18.
1.1.2, 00:07:09
    Traffic Index 8
B
28.
1.4.0/24 [20/0] via
18.
1.1.2, 00:07:09
    Traffic Index 9
B
28.
1.5.0/24 [20/0] via
18.
1.1.2, 00:07:09
    Traffic Index 10
B
65.
0.1.0/24 [20/0] via
4.
1.0.2, 00:07:09
    Traffic Index 15
B
65.
0.2.0/24 [20/0] via
4.
1.0.2, 00:07:09
    Traffic Index 15
B
65.
0.3.0/24 [20/0] via
4.
1.0.2, 00:07:09
    Traffic Index 15
B
65.
0.
4.0/24 [20/0] via
4.
1.0.2, 00:07:09
    Traffic Index 15
B
65.
0.5.0/24 [20/0] via
4.
1.0.2, 00:07:09
    Traffic Index 15
B
65.
0.6.0/24 [20/0] via
4.
1.0.2, 00:07:09
    Traffic Index 15
B
65.
0.7.0/24 [20/0] via
4.
1.0.2, 00:07:09
    Traffic Index 15
```



```
B
65.
0.8.0/24 [20/0] via
4.
1.0.2, 00:07:09
    Traffic Index 15
B
65.
0.9.0/24 [20/0] via
4.
1.0.2, 00:07:09
    Traffic Index 15
B
65.
0.10.0/24 [20/0] via
4.
1.0.2, 00:07:09
    Traffic Index 15
B
66.
0.1.0/24 [20/0] via
8.
1.0.2, 00:07:09
    Traffic Index 16
B
66.
0.2.0/24 [20/0] via
8.
1.0.2, 00:07:09
    Traffic Index 16
B
66.
0.3.0/24 [20/0] via
8.
1.0.2, 00:07:09
    Traffic Index 16
B
66.
0.4.0/24 [20/0] via
8.
1.0.2, 00:07:09
    Traffic Index 16
B
66.
0.5.0/24 [20/0] via
8.
1.0.2, 00:07:09
    Traffic Index 16
B
66.
0.6.0/24 [20/0] via
8.
1.0.2, 00:07:09
    Traffic Index 16
B
66.
0.7.0/24 [20/0] via
8.
1.0.2, 00:07:09
    Traffic Index 16
B
66.
0.8.0/24 [20/0] via
8.
```

```
1.0.2, 00:07:09
    Traffic Index 16
B
66.
0.9.0/24 [20/0] via
8.
1.0.2, 00:07:09
    Traffic Index 16
B
66.
0.10.0/24 [20/0] via
8.
1.0.2, 00:07:09
    Traffic Index 16
B
67.
0.1.0/24 [20/0] via
4.
1.0.2, 00:07:09
    Traffic Index 7
B
67.
0.2.0/24 [20/0] via
4.
1.0.2, 00:07:09
    Traffic Index 7
B
67.
0.3.0/24 [20/0] via
4.
1.0.2, 00:07:09
    Traffic Index 7
B
67.
0.4.0/24 [20/0] via
4.
1.0.2, 00:07:09
    Traffic Index 7
B
67.
0.5.0/24 [20/0] via
4.
1.0.2, 00:07:09
    Traffic Index 7
B
67.
0.6.0/24 [20/0] via
4.
1.0.2, 00:07:09
    Traffic Index 7
B
67.
0.7.0/24 [20/0] via
4.
1.0.2, 00:07:09
    Traffic Index 7
B
67.
0.8.0/24 [20/0] via
4.
1.0.2, 00:07:09
    Traffic Index 7
B
67.
```

```
0.9.0/24 [20/0] via
4.
1.0.2, 00:07:09
    Traffic Index 7
B
    67.
0.10.0/24 [20/0] via
4.
1.0.2, 00:07:09
    Traffic Index 7
B
    68.
0.1.0/24 [20/0] via
8.
1.0.2, 00:07:09
    Traffic Index 8
B
    68.
0.2.0/24 [20/0] via
8.
1.0.2, 00:07:09
    Traffic Index 8
B
    68.
0.3.0/24 [20/0] via
8.
1.0.2, 00:07:09
    Traffic Index 8
B
    68.
0.4.0/24 [20/0] via
8.
1.0.2, 00:07:09
    Traffic Index 8
B
    68.
0.5.0/24 [20/0] via
8.
1.0.2, 00:07:09
    Traffic Index 8
B
    68.
0.6.0/24 [20/0] via
8.
1.0.2, 00:07:09
    Traffic Index 8
B
    68.
0.7.0/24 [20/0] via
8.
1.0.2, 00:07:09
    Traffic Index 8
B
    68.
0.8.0/24 [20/0] via
8.
1.0.2, 00:07:09
    Traffic Index 8
B
    68.
0.9.0/24 [20/0] via
8.
1.0.2, 00:07:09
    Traffic Index 8
```

```

B
  68.
0.10.0/24 [20/0] via
8.
1.0.2, 00:07:09
    Traffic Index 8

show bgp summary

BGP router identifier
190.
1.
1.
1, local AS number 100
BGP generic scan interval 60 secs
BGP main routing table version 151
Dampening enabled
BGP scan interval 60 secs
BGP is operating in STANDALONE mode.

Process          RecvTblVer    bRIB/RIB    SendTblVer
Speaker          151           151         151

Neighbor         Spk   AS  MsgRcvd  MsgSent   TblVer  InQ  OutQ  Up/Down   St/PfxRcd
4.
1.0.2             0   107    54      53      151    0    0  00:25:26    20
8.1.0.2          0   108    54      53      151    0    0  00:25:28    20
17.1.0.2         0   121    53      54      151    0    0  00:25:42     0
17.1.1.2         0   121    53      53      151    0    0  00:25:06     5
17.1.2.2         0   121    52      54      151    0    0  00:25:04     0
17.1.3.2         0   121    52      53      151    0    0  00:25:26     0
17.1.4.2         0   121    53      54      151    0    0  00:25:41     0
17.1.5.2         0   121    53      54      151    0    0  00:25:43     0
17.1.6.2         0   121    51      53      151    0    0  00:24:59     0
17.1.7.2         0   121    51      52      151    0    0  00:24:44     0
17.1.8.2         0   121    51      52      151    0    0  00:24:49     0
18.
1.0.2             0   122    52      54      151    0    0  00:25:21     0
18.
1.1.2             0  1221    54      54      151    0    0  00:25:43     5
18.
1.2.2             0  1222    53      54      151    0    0  00:25:38     0
18.
1.3.2             0  1223    52      53      151    0    0  00:25:17     0
18.
1.4.2             0  1224    51      52      151    0    0  00:24:57     0
18.

```

```

1.5.2          0 1225    52    53    151    0    0 00:25:14    0
18.
1.6.2          0 1226    52    54    151    0    0 00:25:04    0
18.
1.7.2          0 1227    52    54    151    0    0 00:25:13    0
18.
1.8.2          0 1228    53    54    151    0    0 00:25:36    0

```

```
show bgp 27.1.1.1
```

```
BGP routing table entry for 27.1.1.0/24
```

```
Versions:
```

```

  Process          bRIB/RIB  SendTblVer
  Speaker          102      102

```

```
Paths: (1 available, best #1)
```

```
Not advertised to any peer
```

```
Received by speaker 0
```

```
121
```

```
17.1.1.2 from
```

```
17.1.1.2 (
```

```
17.1.1.2)
```

```
Origin incomplete, localpref 100, valid, external, best
```

```
Community: 27:1 121:1
```

```
show bgp
```

```
28.1.1.1
```

```
BGP routing table entry for
```

```
28.1.1.0/24
```

```
Versions:
```

```

  Process          bRIB/RIB  SendTblVer
  Speaker          107      107

```

```
Paths: (1 available, best #1)
```

```
Not advertised to any peer
```

```
Received by speaker 0
```

```
1221
```

```
18.
```

```
1.1.2 from
```

```
18.
```

```
1.1.2 (18.1.1.2)
```

```
Origin incomplete, localpref 100, valid, external, best
```

```
Community: 28:1 1221:1
```

```
show bgp
```

```
65.0.1.1
```

```
BGP routing table entry for
```

```
65.0.1.0/24
```

```
Versions:
```

```

  Process          bRIB/RIB  SendTblVer
  Speaker          112      112

```

```
Paths: (1 available, best #1)
```

```
Not advertised to any peer
```

```
Received by speaker 0
```

```
107
```

```
4.1.0.2 from
```

```
4.1.0.2 (
```

```
4.1.0.2)
```

```

Origin incomplete, localpref 100, valid, external, best
Community: 107:65

show bgp
66.
0.1.1

BGP routing table entry for
66.
0.1.0/24
Versions:
  Process          bRIB/RIB  SendTblVer
  Speaker          122      122
Paths: (1 available, best #1)
  Not advertised to any peer
  Received by speaker 0
  108
    8.1.0.2 from 8.1.0.2 (8.1.0.2)
      Origin incomplete, localpref 100, valid, external, best
      Community: 108:66

show bgp 67.0.1.1

BGP routing table entry for 67.0.1.0/24
Versions:
  Process          bRIB/RIB  SendTblVer
  Speaker          132      132
Paths: (1 available, best #1)
  Not advertised to any peer
  Received by speaker 0
  107
    4.1.0.2 from 4.1.0.2 (4.1.0.2)
      Origin incomplete, localpref 100, valid, external, best
      Community: 107:67

show bgp 68.0.1.1

BGP routing table entry for 68.0.1.0/24
Versions:
  Process          bRIB/RIB  SendTblVer
  Speaker          142      142
Paths: (1 available, best #1)
  Not advertised to any peer
  Received by speaker 0
  108
    8.1.0.2 from 8.1.0.2 (8.1.0.2)
      Origin incomplete, localpref 100, valid, external, best
      Community: 108:68

show route ipv4 27.1.1.1

Routing entry for 27.1.1.0/24
  Known via "bgp 100", distance 20, metric 0
  Tag 121, type external, Traffic Index 1
  Installed Nov 11 21:14:05.462
  Routing Descriptor Blocks
    17.1.1.2, from 17.1.1.2
      Route metric is 0
  No advertising protos.

show route ipv4 28.1.1.1

Routing entry for 28.1.1.0/24
  Known via "bgp 100", distance 20, metric 0

```

```
Tag 1221, type external, Traffic Index 6
Installed Nov 11 21:14:05.462
Routing Descriptor Blocks
  18.1.1.2, from 18.1.1.2
    Route metric is 0
No advertising protos.

show route ipv4 65.0.1.1

Routing entry for 65.0.1.0/24
  Known via "bgp 100", distance 20, metric 0
  Tag 107, type external, Traffic Index 15
  Installed Nov 11 21:14:05.462
  Routing Descriptor Blocks
    4.1.0.2, from 4.1.0.2
      Route metric is 0
  No advertising protos.

show route ipv4 66.0.1.1

Routing entry for 66.0.1.0/24
  Known via "bgp 100", distance 20, metric 0
  Tag 108, type external, Traffic Index 16
  Installed Nov 11 21:14:05.462
  Routing Descriptor Blocks
    8.1.0.2, from 8.1.0.2
      Route metric is 0
  No advertising protos.

show route ipv4 67.0.1.1

Routing entry for 67.0.1.0/24
  Known via "bgp 100", distance 20, metric 0
  Tag 107, type external, Traffic Index 7
  Installed Nov 11 21:14:05.462
  Routing Descriptor Blocks
    4.1.0.2, from 4.1.0.2
      Route metric is 0
  No advertising protos.

show route ipv4 68.0.1.1

Routing entry for 68.0.1.0/24
  Known via "bgp 100", distance 20, metric 0
  Tag 108, type external, Traffic Index 8
  Installed Nov 11 21:14:05.462
  Routing Descriptor Blocks
    8.1.0.2, from 8.1.0.2
      Route metric is 0
  No advertising protos.

show cef ipv4 27.1.1.1

27.1.1.0/24, version 263, source-destination sharing
Prefix Len 24, Traffic Index 1, precedence routine (0)
  via 17.1.1.2, 0 dependencies, recursive
    next hop 17.1.1.2/24, GigabitEthernet 0/2/0/2.1 via 17.1.1.0/24
    valid remote adjacency
  Recursive load sharing using 17.1.1.0/24

show cef ipv4 28.1.1.1

28.1.1.0/24, version 218, source-destination sharing
Prefix Len 24, Traffic Index 6, precedence routine (0)
```

```

via 18.1.1.2, 0 dependencies, recursive
  next hop 18.1.1.2/24, GigabitEthernet0/2/0/4.1 via 18.1.1.0/24
  valid remote adjacency
Recursive load sharing using 18.1.1.0/24

show cef ipv4 65.0.1.1

65.0.1.0/24, version 253, source-destination sharing
Prefix Len 24, Traffic Index 15, precedence routine (0)
  via 4.1.0.2, 0 dependencies, recursive
  next hop 4.1.0.2/16, GigabitEthernet0/0/0/4 via 4.1.0.0/16
  valid remote adjacency
Recursive load sharing using 4.1.0.0/16

show cef ipv4 66.0.1.1

66.0.1.0/24, version 233, source-destination sharing
Prefix Len 24, Traffic Index 16, precedence routine (0)
  via 8.1.0.2, 0 dependencies, recursive
  next hop 8.1.0.2/16, GigabitEthernet 0/0/0/8 via 8.1.0.0/16
  valid remote adjacency
Recursive load sharing using 8.1.0.0/16

show cef ipv4 67.0.1.1

67.0.1.0/24, version 243, source-destination sharing
Prefix Len 24, Traffic Index 7, precedence routine (0)
  via 4.1.0.2, 0 dependencies, recursive
  next hop 4.1.0.2/16, GigabitEthernet 0/0/0/4 via 4.1.0.0/16
  valid remote adjacency
Recursive load sharing using 4.1.0.0/16

show cef ipv4 68.0.1.1

68.0.1.0/24, version 223, source-destination sharing
Prefix Len 24, Traffic Index 8, precedence routine (0)
  via 8.1.0.2, 0 dependencies, recursive
  next hop 8.1.0.2/16, GigabitEthernet0/0/0/8 via 8.1.0.0/16
  valid remote adjacency
Recursive load sharing using 8.1.0.0/16

show cef ipv4 27.1.1.1 detail

27.1.1.0/24, version 263, source-destination sharing
Prefix Len 24, Traffic Index 1, precedence routine (0)
  via 17.1.1.2, 0 dependencies, recursive
  next hop 17.1.1.2/24, GigabitEthernet 0/2/0/2.1 via 17.1.1.0/24
  valid remote adjacency

Recursive load sharing using 17.1.1.0/24
Load distribution: 0 (refcount 6)

Hash OK Interface Address Packets
1 Y GigabitEthernet 0/2/0/2.1 (remote) 0

show cef ipv4 28.1.1.1 detail

28.1.1.0/24, version 218, source-destination sharing
Prefix Len 24, Traffic Index 6, precedence routine (0)
  via 18.1.1.2, 0 dependencies, recursive
  next hop 18.1.1.2/24, GigabitEthernet 0/2/0/4.1 via 18.1.1.0/24
  valid remote adjacency

Recursive load sharing using 18.1.1.0/24

```



```

Load distribution: 0 (refcount 6)

Hash OK Interface Address Packets
1 Y GigabitEthernet 0/2/0/4.1 (remote) 0

show cef ipv4 65.0.1.1 detail

65.0.1.0/24, version 253, source-destination sharing
Prefix Len 24, Traffic Index 15, precedence routine (0)
via 4.1.0.2, 0 dependencies, recursive
next hop 4.1.0.2/16, GigabitEthernet0/0/0/4 via 4.1.0.0/16
valid remote adjacency

Recursive load sharing using 4.1.0.0/16
Load distribution: 0 (refcount 21)

Hash OK Interface Address Packets
1 Y GigabitEthernet0/0/0/4 (remote) 0

show cef ipv4 66.0.1.1 detail

66.0.1.0/24, version 233, source-destination sharing
Prefix Len 24, Traffic Index 16, precedence routine (0)
via 8.1.0.2, 0 dependencies, recursive
next hop 8.1.0.2/16, GigabitEthernet0/0/0/8 via 8.1.0.0/16
valid remote adjacency

Recursive load sharing using 8.1.0.0/16
Load distribution: 0 (refcount 21)

Hash OK Interface Address Packets
1 Y GigabitEthernet 0/0/0/8 (remote) 0

show cef ipv4 67.0.1.1 detail

67.0.1.0/24, version 243, source-destination sharing
Prefix Len 24, Traffic Index 7, precedence routine (0)
via 4.1.0.2, 0 dependencies, recursive
next hop 4.1.0.2/16, GigabitEthernet 0/0/0/4 via 4.1.0.0/16
valid remote adjacency

Recursive load sharing using 4.1.0.0/16
Load distribution: 0 (refcount 21)

Hash OK Interface Address Packets
1 Y GigabitEthernet 0/0/0/4 (remote) 0

show cef ipv4 68.0.1.1 detail

68.0.1.0/24, version 223, source-destination sharing
Prefix Len 24, Traffic Index 8, precedence routine (0)
via 8.1.0.2, 0 dependencies, recursive
next hop 8.1.0.2/16, GigabitEthernet 0/0/0/8 via 8.1.0.0/16
valid remote adjacency

Recursive load sharing using 8.1.0.0/16
Load distribution: 0 (refcount 21)

Hash OK Interface Address Packets
1 Y GigabitEthernet 0/0/0/8 (remote) 0

```

Configuring Unicast RPF Checking: Example

The following example shows how to configure unicast RPF checking:

```
configure
interface GigabitEthernet 0/0/0/1
ipv4 verify unicast source reachable-via rx
end
```

Configuring the Switching of Modular Services Card to Management Ethernet Interfaces on the Route Processor: Example

The following example shows how to configure the switching of the MSC to Management Ethernet interfaces on the route processor:

```
configure
rp mgmtethernet forwarding
end
```

Configuring Per-Flow Load Balancing: Example

The following examples show how to configure Layer 3 and Layer 4 load-balancing for the hash algorithm from the **cef load-balancing fields** command, and how to verify summary information for the CEF table from the **show cef summary** command:

Configuring Layer 3 load-balancing

```
configure
 cef load-balancing fields L3
end
!
show cef summary
Router ID is 10.6.6.6

IP CEF with switching (Table Version 0) for node0_0_CPU0

Load balancing: L3
Tableid 0xe0000000 (0x9cbb51b0), Vrfid 0x60000000, Vrid 0x20000000, Flags 0x2031
Vrfname default, Refcount 577
300 routes, 0 protected, 0 reresolve, 0 unresolved (0 old, 0 new), 21600 bytes
212 load sharing elements, 62576 bytes, 324 references
19 shared load sharing elements, 5388 bytes
193 exclusive load sharing elements, 57188 bytes

622 local route bufs received, 1 remote route bufs received, 0 mix bufs received
176 local routes, 0 remote routes
4096 total local route updates processed
0 total remote route updates processed
0 pkts pre-routed to cust card

0 pkts received from core card
0 CEF route update drops, 96 revisions of existing leaves
0 CEF route update drops due to version mis-match
Resolution Timer: 15s
0 prefixes modified in place
0 deleted stale prefixes
```

```

    82 prefixes with label imposition, 107 prefixes with label information
    95 next hops
    0 incomplete next hops

0 PD backwalks on LDIs with backup path

```

Configuring Layer 4 load-balancing

```

configure
  cef load-balancing fields L4
end
!
show cef summary

Router ID is
10
1.1.1.101

IP CEF with switching (Table Version 0) for node0_RP0_CPU0

Load balancing: L4
Tableid 0xe0000000, Vrfid 0x60000000, Vrid 0x20000000, Flags 0x301
Vrfname default, Refcount 286242
286122 routes, 0 reresolve, 0 unresolved (0 old, 0 new), 20600784 bytes
11124 load sharing elements, 3014696 bytes, 297064 references
8 shared load sharing elements, 3008 bytes
11116 exclusive load sharing elements, 3011688 bytes
0 CEF route update drops, 3900571 revisions of existing leaves
Resolution Timer: 15s
0 prefixes modified in place
0 deleted stale prefixes
0 prefixes with label imposition, 11032 prefixes with label information Adjacency Table
has 15 adjacencies
1 incomplete adjacency

```

Configuring BGP Attributes Download: Example

The following example shows how to configure the BGP Attributes Download feature:

```

router configure
show cef bgp attribute {attribute-id| local-attribute-id}

```

Configuring GTP Tunnel Load Balancing: Example

The following example shows how to enable GTP tunnel load balancing by configuring Layer 4 load-balancing for the 7-tuple hash algorithm:

```

RP/0/RP0/CPU0:router# configure
RP/0/RP0/CPU0:router(config)# cef load-balancing fields L4
RP/0/RP0/CPU0:router(config)# commit

```

The following example shows how to verify summary information for the IPv4 or IPv6 CEF table:

```

RP/0/RP0/CPU0:router# show cef ipv4 summary

```

Router ID is 10.2.2.2

IP CEF with switching (Table Version 0) for node0_RP0_CPU0

```

Load balancing: L4
Tableid 0xe0000000 (0x9cdbcldc), Vrfid 0x60000000, Vrid 0x20000000, Flags 0x21
Vrfname default, Refcount 527
293 routes, 0 protected, 0 reresolve, 0 unresolved (0 old, 0 new), 23440 bytes
222 load sharing elements, 64376 bytes, 85 references
14 shared load sharing elements, 4064 bytes
208 exclusive load sharing elements, 60312 bytes
0 route delete cache elements
2036 local route bufs received, 1264 remote route bufs received, 0 mix bufs d
117 local routes, 0 remote routes
8762 total local route updates processed
0 total remote route updates processed
0 pkts pre-routed to cust card
0 pkts pre-routed to rp card
0 pkts received from core card
0 CEF route update drops, 2151 revisions of existing leaves
0 CEF route update drops due to version mis-match
Resolution Timer: 15s
0 prefixes modified in place
0 deleted stale prefixes
0 prefixes with label imposition, 0 prefixes with label information
0 LISP EID prefixes, 0 merged, via 0 rlocs
159 next hops
0 incomplete next hops

0 PD backwalks on LDIs with backup path

```

RP/0/RP0/CPU0:router# **show cef ipv6 summary**

Router ID is 10.2.2.2

IP CEF with switching (Table Version 0) for node0_RP0_CPU0

```

Load balancing: L4
Tableid 0xe0800000 (0x9cdee368), Vrfid 0x60000000, Vrid 0x20000000, Flags 0x21
Vrfname default, Refcount 39
17 routes, 0 protected, 0 reresolve, 0 unresolved (0 old, 0 new), 1360 bytes
17 load sharing elements, 4876 bytes, 4 references
4 shared load sharing elements, 1072 bytes
13 exclusive load sharing elements, 3804 bytes
0 route delete cache elements
199321 local route bufs received, 49838 remote route bufs received, 0 mix bud
9 local routes, 0 remote routes
1046420 total local route updates processed
0 total remote route updates processed
0 pkts pre-routed to cust card
0 pkts pre-routed to rp card
0 pkts received from core card
0 CEF route update drops, 1 revisions of existing leaves
0 CEF route update drops due to version mis-match
Resolution Timer: 15s
0 prefixes modified in place
0 deleted stale prefixes
0 prefixes with label imposition, 0 prefixes with label information
0 LISP EID prefixes, 0 merged, via 0 rlocs
3 next hops
0 incomplete next hops

```

0 PD backwalks on LDIs with backup path

Use the **show cef {ipv4 | ipv6} exact-route** command in EXEC mode to verify that the interface selected for load balancing matches with the output from this command. The following examples show the exact route for the Layer 4 information:



Note Use 16MSBist of TEID in source port and 16LSBits of TEID in destination port in place of L4 protocol source and destination port number.

For example:

If TEID=241210E1 (in hexadecimal), then source-port=9234 (Decimal equivalent of 16MSBits of TEID 2412) and destination-port=4321 (Decimal equivalent of 16LSBits of TEID 10E1)

If TEID=0069012F (in hexadecimal), then source-port= 105 (Decimal equivalent of 16MSBits of TEID 0069) and destination-port= 303 (Decimal equivalent of 16LSBits of TEID 012F)

```
RP/0/RP0/CPU0:router# show cef ipv4 exact-route 20.0.0.2 60.0.0.2 protocol udp source-port
9234 destination-port 4321 ingress-interface GigabitEthernet 0/6/5/0
```

```
0.0.0.0/0, version 12, proxy default, internal 0x4000021 (ptr 0x9d760060) [1], )
Updated Jul 17 03:12:35.566
local adjacency 172.29.52.1
Prefix Len 0, traffic index 0, precedence routine (0), priority 3
via MgmtEth0/RP0/CPU0/0
via 172.29.52.1, 5 dependencies, recursive [flags 0x0]
path-idx 0 [0x9d760648 0x0]
next hop 172.29.52.1 via 172.29.52.1/32
```

```
RP/0/RP0/CPU0:router# show cef ipv6 exact-route 20:6:1::9 22:6:1::9 protocol udp source-port
105 destination-port 303 ingress-interface GigabitEthernet 0/6/5/4
```

```
::/0, version 8, proxy default, internal 0x4000021 (ptr 0x9d6ac06c) [1], 0x0 (0)
Updated Jul 17 03:14:49.695
remote adjacency to GigabitEthernet0/6/5/0.22
Prefix Len 0, traffic index 0, precedence routine (0), priority 3
via GigabitEthernet0/6/5/0.22
via 5001:db8::1, GigabitEthernet0/6/5/0.22, 4 dependencies, weight 0, class ]
path-idx 0 [0x9dd7e0c4 0x0]
next hop 5001:db8::1
remote adjacency
```

Additional References

The following sections provide references related to implementing CEF.

Related Documents

| Related Topic | Document Title |
|---|---|
| CEF commands: complete command syntax, command modes, command history, defaults, usage guidelines, and examples | <i>Cisco Express Forwarding Commands</i> module in <i>IP Addresses and Services Command Reference for Cisco CRS Routers</i> |

| Related Topic | Document Title |
|---|---|
| BGP commands: complete command syntax, command modes, command history, defaults, usage guidelines, and examples | <i>BGP Commands</i> module in the <i>Routing Command Reference for Cisco CRS Routers</i> |
| Link Bundling Commands: complete command syntax, command modes, command history, defaults, usage guidelines, and examples | <i>Link Bundling Commands</i> module in the <i>Interface and Hardware Component Command Reference for Cisco CRS Routers</i> |

Standards

| Standards | Title |
|---|-------|
| No new or modified standards are supported by this feature, and support for existing standards has not been modified by this feature. | — |

MIBs

| MIBs | MIBs Link |
|------|---|
| — | To locate and download MIBs, use the Cisco MIB Locator found at the following URL and choose a platform under the Cisco Access Products menu: https://mibs.cloudapps.cisco.com/ITDIT/MIBS/servlet/index |

RFCs

| RFCs | Title |
|---|-------|
| No new or modified RFCs are supported by this feature, and support for existing RFCs has not been modified by this feature. | — |

Technical Assistance

| Description | Link |
|---|---|
| The Cisco Technical Support website contains thousands of pages of searchable technical content, including links to products, technologies, solutions, technical tips, and tools. Registered Cisco.com users can log in from this page to access even more content. | http://www.cisco.com/techsupport |



CHAPTER 9

Implementing EPFT

The Excessive Punt Flow Trap (EPFT) feature attempts to identify and mitigate control packet traffic from remote devices that send more than their allocated share of control packet traffic. A remote device can be a device on a VLAN interface or a device identified by its source MAC address. When remote devices send control packet traffic to the router, the control packets are punted and policed by a local packet transport service (LPTS) queue to protect the router's CPU. If one device sends an excessive rate of control packet traffic, the policer queue fills up, causing many packets to be dropped. If the rate from one "bad actor" device greatly exceeds that of other devices, most of the other devices do not get any of their control packets through to the router. The Excessive Punt Flow Trap feature addresses this situation.

For a complete description of the EPFT commands listed in this module, refer to the EPFT Commands module of *Cisco IOS XR IP Addresses and Services Configuration Guide for the Cisco CRS Router*.

Feature History for Implementing EPFT

| Release | Modification |
|---------------|--|
| Release 5.3.2 | EPFT feature support on Cisco CRS-3 Modular Services Line Card was introduced. |
| Release 5.3.1 | The EPFT configuration feature was introduced. |

- [Information About EPFT, on page 175](#)
- [Prerequisites for Implementing EPFT, on page 176](#)
- [Restrictions for Implementing EPFT, on page 176](#)
- [Enabling Excessive Punt Flow Trap Processing, on page 176](#)
- [Additional References, on page 178](#)

Information About EPFT

The Excessive Punt Flow Trap feature monitors control packet traffic arriving from physical interfaces, sub-interfaces, bundle interfaces, and bundle sub-interfaces. If the source that floods the punt queue with packets is a device with an interface handle, then all punts from that bad actor interface are penalty policed.

The EPFT feature supports policing the bad actors for Address Resolution Protocol (ARP). ARP has a static punt rate and a penalty rate. For example, the sum total of all ARP punts from remote devices is policed at 1000 packets per second (pps) to the router's CPU. If one remote device sends an excessive rate of ARP traffic and is trapped, then ARP traffic from that bad actor is policed at 10 pps. The remaining (non-bad) remote devices continue to use the static 1000 pps queue for ARP. The excessive rate required to cause an interface

to get trapped has nothing to do with the static punt rate (for example, 1000 pps for ARP). The excessive rate is a rate that is significantly higher than the current average rate of other control packets being punted. The excessive rate is not a fixed rate, and is dependent on the current overall punt packet activity.

Once a bad actor is trapped, it is penalty policed on its punted protocol (ARP), irrespective of the protocol that caused it to be identified as a bad actor. A penalty rate of 10 pps is sufficient to allow the other protocols to function normally. When an interface is trapped, it is placed in a "penalty box" for a period of time (a default of 15 minutes). At the end of the penalty timeout, it is removed from penalty policing (or dropping). If there is still an excessive rate of control packet traffic coming from the remote device, then the interface is trapped again.

For more information about enabling the Excessive Punt Flow Trap feature, see [Enabling Excessive Punt Flow Trap Processing, on page 176](#).



Note Even when the Excessive Punt Flow Trap feature is not enabled, the "bad actors" can affect services for only other devices; they cannot bring down the router.

Prerequisites for Implementing EPFT

The following prerequisites are required to implement EPFT:

- The EPFT feature is supported on Cisco CRS-3 Modular Services Line Card and Cisco CRS-X Line Card.
- You must be in a user group associated with a task group that includes the proper task IDs. The command reference guides include the task IDs required for each command. If you suspect user group assignment is preventing you from using a command, contact your AAA administrator for assistance.

Restrictions for Implementing EPFT

These restrictions apply to implementing EPFT feature:

- This feature is non-deterministic. In some cases, the EPFT feature can give a false positive, i.e. it could trap an interface that is sending legitimate punt traffic.
- The EPFT feature traps flows based on the relative rate of different flows; thus, the behavior depends on the ambient punt rates. A flow that is significantly higher than other flows could be trapped as a bad actor. Thus the feature is less sensitive when there are many flows, and more sensitive when there are fewer flows present.
- Sometimes control packet traffic can occur in bursts. EPFT has safeguards against triggering on short bursts, but longer bursts could trigger a false positive trap.

Enabling Excessive Punt Flow Trap Processing

Perform this task to enable the Excessive Punt Flow Trap (EPFT) feature. The task also enables you to set the penalty policing rate and penalty timeout for a protocol and exclude EPFT processing on a specific interface.

SUMMARY STEPS

1. **configure**
2. **lpts punt excessive-flow-trap non-subscriber-interfaces**
3. **lpts punt excessive-flow-trap penalty-rate** *protocol penalty_policer_rate*
4. **lpts punt excessive-flow-trap penalty-timeout** *protocol time*
5. **lpts punt excessive-flow-trap exclude interface** *interface-name*
6. **commit**

DETAILED STEPS

| | Command or Action | Purpose |
|--------|---|--|
| Step 1 | configure | |
| Step 2 | lpts punt excessive-flow-trap non-subscriber-interfaces Example: RP/0/RP0/CPU0:router(config)# lpts punt excessive-flow-trap non-subscriber-interfaces | Enables the Excessive Punt Flow Trap feature on non-subscriber interfaces. |
| Step 3 | lpts punt excessive-flow-trap penalty-rate <i>protocol penalty_policer_rate</i> Example: RP/0/RP0/CPU0:router(config)# lpts punt excessive-flow-trap penalty-rate arp 10 | Sets the penalty policing rate for a protocol. The penalty policer rate is in packets-per-second (pps) and ranges from 2 to 100. Note The penalty policing rate for a protocol consumes a policer rate profile. |
| Step 4 | lpts punt excessive-flow-trap penalty-timeout <i>protocol time</i> Example: RP/0/RP0/CPU0:router(config)# lpts punt excessive-flow-trap penalty-timeout arp 10 | Sets the penalty timeout value, which is a period of time that the interface trapped is placed in the penalty box, for a protocol. The penalty timeout value is in minutes and ranges from 1 to 1000. The default penalty timeout value is 15 minutes. |
| Step 5 | lpts punt excessive-flow-trap exclude interface <i>interface-name</i> Example: RP/0/RP0/CPU0:router(config)# lpts punt excessive-flow-trap exclude interface GigabitEthernet0/6/0/6.111 | Excludes EPFT processing on a specific interface. |
| Step 6 | commit | |

Enabling Excessive Punt Flow Trap Processing: Example

This is an example for enabling the Excessive Punt Flow Trap for non-subscriber interfaces, using the default penalty rate (10 pps) and setting the ARP penalty timeout to 2 minutes.

```
configure
lpts punt excessive-flow-trap non-subscriber-interfaces
lpts punt excessive-flow-trap penalty-rate arp 10
```

```
lpts punt excessive-flow-trap penalty-timeout arp 2
lpts punt excessive-flow-trap exclude interface GigabitEthernet0/6/0/6.111
end
!!
```

Use any of the following **show** commands in EXEC mode to display information about bad actors, penalty status, and other details about the Excessive Punt Flow Trap feature:

- **show lpts punt excessive-flow-trap** [*protocol*]
- **show lpts punt excessive-flow-trap interface** *interface-name* [*protocol*]
- **show running lpts punt excessive-flow-trap**
- **show lpts punt excessive-flow-trap information**

Additional References

The following sections provide references related to implementing Network Stack IPv4 and IPv6.

Related Documents

| Related Topic | Document Title |
|---|---|
| Address resolution configuration tasks | <i>Configuring ARP</i> module in this publication. |
| Mapping host names to IP addresses | <i>Host Services and Applications Commands</i> module in the <i>IP Addresses and Services Command Reference for Cisco CRS Routers</i> |
| Network stack IPv4 and IPv6 commands: complete command syntax, command modes, command history, defaults, usage guidelines, and examples | <i>Network Stack IPv4 and IPv6 Commands</i> section in the <i>IP Addresses and Services Command Reference for Cisco CRS Routers</i> |

Standards

| Standards | Title |
|---|-------|
| No new or modified standards are supported by this feature, and support for existing standards has not been modified by this feature. | — |

MIBs

| MIBs | MIBs Link |
|------|--|
| — | To locate and download MIBs using Cisco IOS XR software, use the Cisco MIB Locator found at the following URL and choose a platform under the Cisco Access Products menu: https://mibs.cloudapps.cisco.com/ITDIT/MIBS/servlet/index |

RFCs

| RFCs | Title |
|---|-------|
| No new or modified RFCs are supported by this feature, and support for existing RFCs has not been modified by this feature. | — |

Technical Assistance

| Description | Link |
|---|---|
| The Cisco Technical Support website contains thousands of pages of searchable technical content, including links to products, technologies, solutions, technical tips, and tools. Registered Cisco.com users can log in from this page to access even more content. | http://www.cisco.com/techsupport |



CHAPTER 10

Implementing HSRP

The Hot Standby Router Protocol (HSRP) is an IP routing redundancy protocol designed to allow for transparent failover at the first-hop IP router. HSRP provides high network availability, because it routes IP traffic from hosts on networks without relying on the availability of any single router. HSRP is used in a group of routers for selecting an active router and a standby router. (An active router is the router of choice for routing packets; a standby router is a router that takes over the routing duties when an active router fails, or when preset conditions are met.)

Feature History for Implementing HSRP

| Release | Modification |
|---------------|--|
| Release 2.0 | This feature was introduced. |
| Release 3.4.0 | This feature was updated to support the minimum and reload delay options. |
| Release 3.5.0 | HSRP supports Ethernet link bundles. |
| Release 3.9.0 | <ul style="list-style-type: none">• BFD for HSRP feature was added.• Hot restartability for HSRP feature was added. |
| Release 4.2.0 | Multiple Group Optimization (MGO) for HSRP feature was added. |
| Release 4.2.1 | Enhanced object tracking for HSRP and IP Static feature was added. |



Note GLBP is not supported on ASR9k.

- [Prerequisites for Implementing HSRP](#) , on page 182
- [Restrictions for Implementing HSRP](#) , on page 182
- [Information About Implementing HSRP](#) , on page 182
- [How to Implement HSRP](#) , on page 185
- [BFD for HSRP](#) , on page 205
- [Enhanced Object Tracking for HSRP and IP Static](#) , on page 208
- [Hot Restartability for HSRP](#) , on page 210
- [Configuration Examples for HSRP Implementation on Software](#) , on page 210

- [Additional References, on page 211](#)

Prerequisites for Implementing HSRP

You must be in a user group associated with a task group that includes the proper task IDs. The command reference guides include the task IDs required for each command. If you suspect user group assignment is preventing you from using a command, contact your AAA administrator for assistance.

Restrictions for Implementing HSRP

HSRP is supported on Ethernet interfaces, Ethernet sub-interfaces, Ethernet link bundles, and Bridge Virtual Interfaces (BVI).

Information About Implementing HSRP

To implement HSRP on Cisco IOS XR software software, you need to understand the following concepts:

HSRP Overview

HSRP is useful for hosts that do not support a router discovery protocol (such as Internet Control Message Protocol [ICMP] Router Discovery Protocol [IRDP]) and cannot switch to a new router when their selected router reloads or loses power. Because existing TCP sessions can survive the failover, this protocol also provides a more transparent recovery for hosts that dynamically choose a next hop for routing IP traffic.

When HSRP is configured on a network segment, it provides a virtual MAC address and an IP address that is shared among a group of routers running HSRP. The address of this HSRP group is referred to as the *virtual IP address*. One of these devices is selected by the protocol to be the *active router*. The active router receives and routes packets destined for the MAC address of the group. For n routers running HSRP, $n + 1$ IP and MAC addresses are assigned.

HSRP detects when the designated active router fails, at which point a selected standby router assumes control of the MAC and IP addresses of the HSRP group. A new *standby router* is also selected at that time.

Devices that are running HSRP send and receive multicast User Datagram Protocol (UDP) based hello packets to detect router failure and to designate active and standby routers.

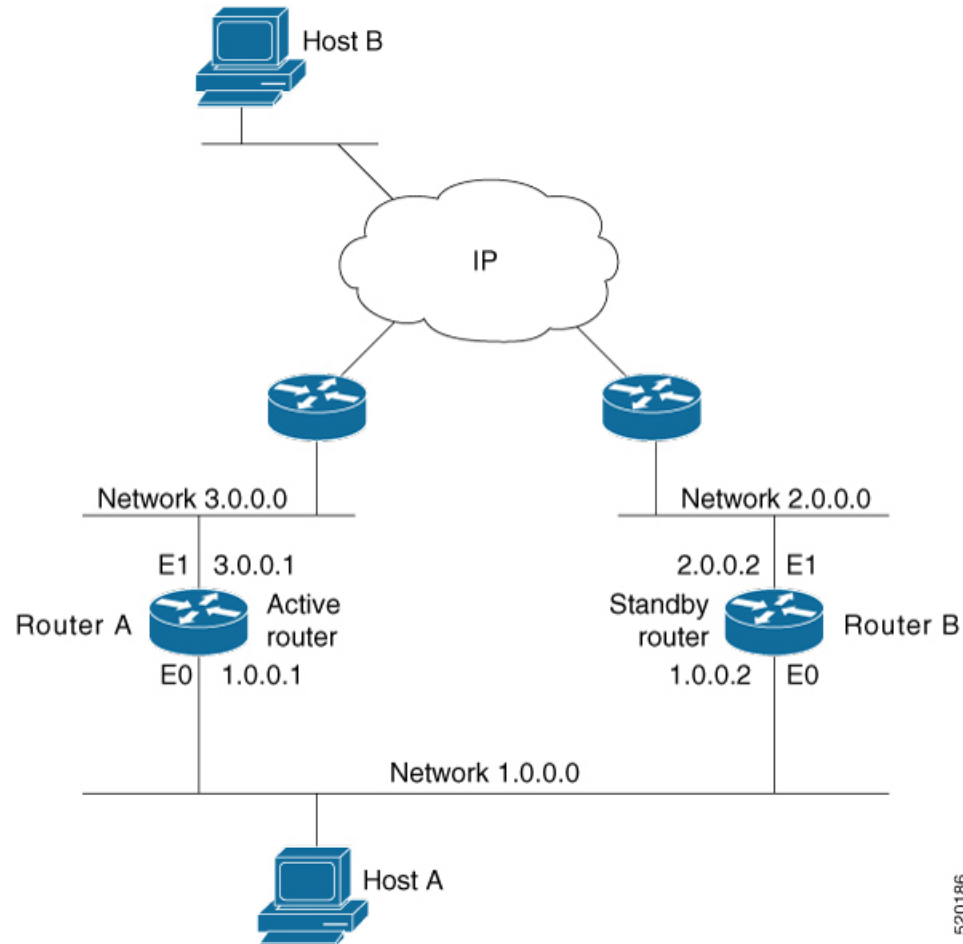
HSRP Groups

An HSRP group consists of two or more routers running HSRP that are configured to provide hot standby services for one another. HSRP uses a priority scheme to determine which HSRP-configured router is to be the default active router. To configure a router as the active router, you assign it a priority that is higher than the priority of all the other HSRP-configured routers. The default priority is 100, so if you configure just one router to have a higher priority, that router will be the default active router.

HSRP works by the exchange of multicast messages that advertise priority among the HSRP group. When the active router fails to send a hello message within a configurable period of time, the standby router with the highest priority becomes the active router. The transition of packet-forwarding functions between routers is completely transparent to all hosts on the network.

Figure 14: Routers Configured as an HSRP Group, on page 183 shows routers configured as members of a single HSRP group.

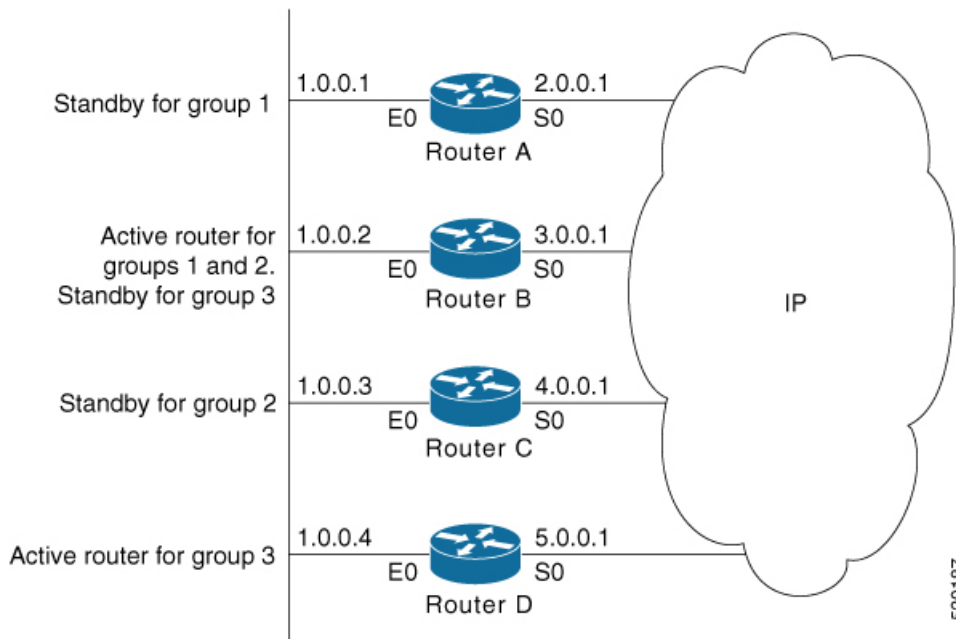
Figure 14: Routers Configured as an HSRP Group



All hosts on the network are configured to use the IP address of the virtual router (in this case, 1.0.0.3) as the default gateway.

A single router interface can also be configured to belong to more than one HSRP group. Figure 15: Routers Configured as Members of Multiple HSRP Groups, on page 184 shows routers configured as members of multiple HSRP groups.

Figure 15: Routers Configured as Members of Multiple HSRP Groups



In Figure 15: Routers Configured as Members of Multiple HSRP Groups, on page 184, the Ethernet interface 0 of Router A belongs to group 1. Ethernet interface 0 of Router B belongs to groups 1, 2, and 3. The Ethernet interface 0 of Router C belongs to group 2, and the Ethernet interface 0 of Router D belongs to group 3. When you establish groups, you might want to align them along departmental organizations. In this case, group 1 might support the Engineering Department, group 2 might support the Manufacturing Department, and group 3 might support the Finance Department.

Router B is configured as the active router for groups 1 and 2 and as the standby router for group 3. Router D is configured as the active router for group 3. If Router D fails for any reason, Router B assumes the packet-transfer functions of Router D and maintains the ability of users in the Finance Department to access data on other subnets.



Note A different virtual MAC address (VMAC) is required for each sub interface. VMAC is determined from the group ID. Therefore, a unique group ID is required for each sub interface configured, unless the VMAC is configured explicitly.



Note We recommend that you disable Spanning Tree Protocol (STP) on switch ports to which the virtual routers are connected. Enable RSTP or rapid-PVST on the switch interfaces if the switch supports these protocols.

HSRP and ARP

When a router in an HSRP group goes active, it sends a number of ARP responses containing its virtual IP address and the virtual MAC address. These ARP responses help switches and learning bridges update their

port-to-MAC maps. These ARP responses also provide routers configured to use the burned-in address of the interface as its virtual MAC address (instead of the preassigned MAC address or the functional address) with a means to update the ARP entries for the virtual IP address. Unlike the gratuitous ARP responses sent to identify the interface IP address when an interface comes up, the HSRP router ARP response packet carries the virtual MAC address in the packet header. The ARP data fields for IP address and media address contain the virtual IP and virtual MAC addresses.

Preemption

The HSRP preemption feature enables the router with highest priority to immediately become the active router. Priority is determined first by the priority value that you configure, and then by the IP address. In each case, a higher value is of greater priority.

When a higher-priority router preempts a lower-priority router, it sends a coup message. When a lower-priority active router receives a coup message or hello message from a higher-priority active router, it changes to the speak state and sends a resign message.

ICMP Redirect Messages

Internet Control Message Protocol (ICMP) is a network layer Internet protocol that provides message packets to report errors and other information relevant to IP processing. ICMP provides many diagnostic functions and can send and redirect error packets to the host. When running HSRP, it is important to prevent hosts from discovering the interface (or real) MAC addresses of routers in the HSRP group. If a host is redirected by ICMP to the real MAC address of a router, and that router later fails, then packets from the host are lost.

ICMP redirect messages are automatically enabled on interfaces configured with HSRP. This functionality works by filtering outgoing ICMP redirect messages through HSRP, where the next-hop IP address may be changed to an HSRP virtual IP address.

To support ICMP redirects, redirect messages are filtered through HSRP, where the next-hop IP address is changed to an HSRP virtual address. When HSRP redirects are turned on, ICMP interfaces with HSRP do this filtering. HSRP keeps track of all HSRP routers by sending advertisements and maintaining a real IP address to virtual IP address mapping to perform the redirect filtering.

How to Implement HSRP

This section contains instructions for the following tasks:

Enabling HSRP

The **hsrp ipv4** 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 virtual address is learned from the active router. For HSRP to elect a designated router, at least one router in the Hot Standby group must have been configured with, or learned, the designated address. Configuring the designated address on the active router always overrides a designated address that is currently in use.

SUMMARY STEPS

1. **configure**
2. **router hsrp**

3. **interface** type interface-path-id
4. **address-family ipv4**
5. **hsrp group-number version version-no**
6. **address { learn | address [secondary] }**
7. **commit**

DETAILED STEPS

| | Command or Action | Purpose |
|---------------|---|---|
| Step 1 | configure | |
| Step 2 | router hsrp Example: RP/0/RP0/CPU0:router(config)# router hsrp | Enables HSRP configuration mode. |
| Step 3 | interface type interface-path-id Example: RP/0/RP0/CPU0:router(config-hsrp)# interface GigabitEthernet 0/2/0/1 | Enables HSRP interface configuration mode on a specific interface. |
| Step 4 | address-family ipv4 Example: RP/0/RP0/CPU0:router(config-hsrp-if)# address-family ipv4 | Enables HSRP address-family configuration mode on a specific interface. |
| Step 5 | hsrp group-number version version-no Example: RP/0/RP0/CPU0:router(config-hsrp-ipv4)# hsrp 1 version 1 | Enables HSRP group submode. Note The version keyword is available only if IPv4 address-family is selected. By default, version is set to 2 for IPv6 address families. |
| Step 6 | address { learn address [secondary] } Example: RP/0/RP0/CPU0:router(config-hsrp-gp)# address learn | Activates HSRP on the configured interface. <ul style="list-style-type: none"> • 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 virtual address is learned from the active router. Note If you configure HSRP for IPv6, you must configure a link local IPv6 address or enable it using the autoconfig keyword. If you do not configure a linklocal IPv6 address, the router does not accept the configuration when you commit your changes using the commit keyword. |
| Step 7 | commit | |

Enabling HSRP for IPv6

Use the following steps to enable HSRP for IPv6.

SUMMARY STEPS

1. **configure**
2. **router hsrp**
3. **interface** *type interface-path-id*
4. **address-family ipv6**
5. **hsrp group-number**
6. **address linklocal** {**autoconfig** | *ipv6-address*}
7. **address global** *ipv6-address*
8. **commit**

DETAILED STEPS

| | Command or Action | Purpose |
|--------|---|--|
| Step 1 | configure | |
| Step 2 | router hsrp Example: RP/0/RP0/CPU0:router(config)# router hsrp | Enables HSRP configuration mode. |
| Step 3 | interface <i>type interface-path-id</i> Example: RP/0/RP0/CPU0:router(config-hsrp)# interface TenGigE 0/2/0/1 | Enables HSRP interface configuration mode on a specific interface. |
| Step 4 | address-family ipv6 Example: RP/0/RP0/CPU0:router(config-hsrp-if)# address-family ipv6 | Enables HSRP address-family configuration mode on a specific interface. |
| Step 5 | hsrp group-number Example: RP/0/RP0/CPU0:router(config-hsrp-ipv4)# hsrp 1 | Enables HSRP group submenu. Note The version keyword is available only if IPv4 address-family is selected. By default, version is set to 2 for IPv6 address families. |
| Step 6 | address linklocal { autoconfig <i>ipv6-address</i> } Example: RP/0/RP0/CPU0:router(config-hsrp-gp)# address linklocal autoconfig | Activates HSRP on the configured interface and assigns a linklocal IPv6 address. • The virtual linklocal address must not match any other virtual linklocal address that is already configured for a different group. |

| | Command or Action | Purpose |
|---------------|---|---|
| | | <p>The virtual linklocal address must not match the interface linklocal IPv6 address.</p> <p>If you use the autoconfig keyword, the linklocal address is calculated using the EUI-64 format.</p> <p>Use the legacy-compatible keyword to be compatible with Cisco IOS and other legacy Cisco devices.</p> |
| Step 7 | <p>address global <i>ipv6-address</i></p> <p>Example:</p> <pre>RP/0/RP0/CPU0:router(config-hsrp-gp)# address global 2001:DB8:A:B::1</pre> | <p>Activates HSRP on the configured interface and assigns a global IPv6 address.</p> <p>Note If you configure HSRP for IPv6, you must configure a link local IPv6 address or enable it using the autoconfig keyword. If you do not configure a linklocal IPv6 address, the router does not accept the configuration when you commit your changes using the commit keyword.</p> |
| Step 8 | commit | |

Configuring HSRP Group Attributes

To configure other Hot Standby group attributes that affect how the local router participates in HSRP, use the following procedure in interface configuration mode as needed:

SUMMARY STEPS

1. **configure**
2. **router hsrp**
3. **interface** *type interface-path-id*
4. **hsrp use-bia**
5. **address-family ipv4**
6. **hsrp** *group-number* **version** *version-no*
7. **priority** *priority*
8. **track** *type instance* [*priority-decrement*]
9. **preempt** [*delay seconds*]
10. **authentication** *string*
11. **mac-address** *address*
12. **commit**

DETAILED STEPS

| | Command or Action | Purpose |
|---------------|-------------------|---------|
| Step 1 | configure | |

| | Command or Action | Purpose |
|---------------|---|---|
| Step 2 | router hsrp Example: RP/0/RP0/CPU0:router(config)# router hsrp | Enables HSRP configuration mode. |
| Step 3 | interface type interface-path-id Example: RP/0/RP0/CPU0:router(config-hsrp)# interface TenGigE 0/2/0/1 | Enables HSRP interface configuration mode on a specific interface. |
| Step 4 | hsrp use-bia Example: RP/0/RP0/CPU0:router(config-hsrp-if)# hsrp use-bia | (Optional) Configures the HSRP to use the burned-in address of the interface as its virtual MAC address, instead of the preassigned MAC address or the functional address. <ul style="list-style-type: none"> • Enter the use-bia command on an interface when there are devices that reject Address Resolution Protocol (ARP) replies with source hardware addresses set to a functional address. • To restore the default virtual MAC address, use the no hsrp use-bia command. |
| Step 5 | address-family ipv4 Example: RP/0/RP0/CPU0:router(config-hsrp-if)# address-family ipv4 | Enables HSRP address-family configuration mode on a specific interface. |
| Step 6 | hsrp group-number version version-no Example: RP/0/RP0/CPU0:router(config-hsrp-ipv4)# hsrp 1 version 1 | Enables HSRP group submenu. <p>Note The version keyword is available only if IPv4 address-family is selected. By default, version is set to 2 for IPv6 address families.</p> |
| Step 7 | priority priority Example: RP/0/RP0/CPU0:router(config-hsrp-gp)# priority 100 | (Optional) Configures HSRP priority. <ul style="list-style-type: none"> • The assigned priority is used to help select the active and standby routers. Assuming that preemption is enabled, the router with the highest priority becomes the designated active router. In case of ties, the primary IP addresses are compared, and the higher IP address has priority. • The priority of the device can change dynamically if an interface is configured with the track command and another interface on the device goes down. • If preemption is not enabled using the preempt command, the router may not become active even |

| | Command or Action | Purpose |
|---------------|---|---|
| | | <p>though it might have a higher priority than other HSRP routers.</p> <ul style="list-style-type: none"> To restore the default HSRP priority values, use the no priority command. |
| Step 8 | <p>track <i>type</i> instance [<i>priority-decrement</i>]</p> <p>Example:</p> <pre>RP/0/RP0/CPU0:router(config-hsrp-gp)# track TenGigE 0/3/0/1</pre> | <p>(Optional) Configures an interface so that the Hot Standby priority changes on the basis of the availability of other interfaces.</p> <ul style="list-style-type: none"> When a tracked interface goes down, the Hot Standby priority decreases by 10. If an interface is not tracked, its state changes do not affect the Hot Standby priority. For each interface configured for Hot Standby, you can configure a separate list of interfaces to be tracked. The optional <i>priority-decrement</i> argument specifies by how much to decrement the Hot Standby priority when a tracked interface goes down. When the tracked interface comes back up, the priority is incrementally increased by the same amount. When multiple tracked interfaces are down and the <i>priority-decrement</i> argument has been configured, these configured priority decrements are cumulative. If tracked interfaces are down, but none of them were configured with priority decrements, the default decrement is 10 and it is cumulative. The preempt command must be used in conjunction with this command on all routers in the group whenever the best available router should be used to forward packets. If the preempt command is not used, the active router stays active, regardless of the current priorities of the other HSRP routers. To remove the tracking, use the no preempt command. |
| Step 9 | <p>preempt [<i>delay seconds</i>]</p> <p>Example:</p> <pre>RP/0/RP0/CPU0:router(config-hsrp-gp)# preempt</pre> | <p>(Optional) Configures HSRP preemption and preemption delay.</p> <ul style="list-style-type: none"> When you configure preemption and preemption delay with the preempt command, the local router attempts to assume control as the active router when the local router has a Hot Standby priority higher than the current active router. If the preempt command is not configured, the local router assumes control as the active router only if it receives information indicating that no router is currently in the active state (acting as the designated router). |

| | Command or Action | Purpose |
|-----------------------|---|--|
| | | <ul style="list-style-type: none"> • When a router first comes up, it does not have a complete routing table. If it is configured to preempt, it becomes the active router, yet it is unable to provide adequate routing services. This problem can be solved by configuring a delay before the preempting router actually preempts the currently active router. • The preempt <i>delay seconds</i> value does not apply if there is no router currently in the active state. In this case, the local router becomes active after the appropriate timeouts (see the timers command), regardless of the preempt delay seconds value. • To restore the default HSRP preemption and preemption delay values, use the no preempt command. |
| <p>Step 10</p> | <p>authentication <i>string</i></p> <p>Example:</p> <pre>RP/0/RP0/CPU0:router(config-hsrp-gp)# authentication companyl</pre> | <p>(Optional) Configures an authentication string for the Hot Standby Router Protocol (HSRP).</p> <ul style="list-style-type: none"> • The authentication string is sent unencrypted in all HSRP messages. The same authentication string must be configured on all routers and access servers on a LAN to ensure interoperation. • Authentication mismatch prevents a device from learning the designated Hot Standby IP address and the Hot Standby timer values from other routers configured with HSRP. • Authentication mismatch does not prevent protocol events such as one router taking over as the designated router. • To delete an authentication string, use the no authentication command. |
| <p>Step 11</p> | <p>mac-address <i>address</i></p> <p>Example:</p> <pre>RP/0/RP0/CPU0:router(config-hsrp-if)# mac-address 4000.1000.1060</pre> | <p>(Optional) Specifies a virtual MAC address for the HSRP.</p> <ul style="list-style-type: none"> • We do not recommend this command, except for IBM networking environments in which first-hop redundancy is based on being able to use a virtual MAC address, and in which you cannot change the first-hop addresses in the PCs that are connected to an Ethernet switch. • HSRP is used to help end stations locate the first-hop gateway for IP routing. The end stations are configured with a default gateway. However, HSRP can provide first-hop redundancy for other protocols. Some protocols, such as Advanced Peer-to-Peer Networking (APPN), use the MAC address to identify |

| | Command or Action | Purpose |
|----------------|-------------------|--|
| | | <p>the first-hop for routing purposes. In this case, it is often necessary to specify the virtual MAC address; the virtual IP address is unimportant for these protocols. Use the mac-address command to specify the virtual MAC address.</p> <ul style="list-style-type: none"> • The MAC address specified is used as the virtual MAC address when the router is active. • The mac-address command is intended for certain APPN configurations. • In an APPN network, an end node is typically configured with the MAC address of the adjacent network node. Use the mac-address command in the routers to set the virtual MAC address to the value used in the end nodes. • Enter the no mac-address command to revert to the standard virtual MAC address (0000.0C07.ACn). |
| Step 12 | commit | |

Configuring the HSRP Activation Delay

The activation delay for HSRP is designed to delay the startup of the state machine when an interface comes up. This give the network time to settle and avoids unnecessary state changes early after the link comes up.

SUMMARY STEPS

1. **configure**
2. **router hsrp**
3. **interface** *type interface-path-id*
4. **hsrp delay** [**minimum** *seconds*] [**reload** *seconds*]
5. **address-family ipv4**
6. **hsrp** *group-number* **version** *version-no*
7. **address** { **learn** | *address* [**secondary**] }
8. **commit**

DETAILED STEPS

| | Command or Action | Purpose |
|---------------|---|----------------------------------|
| Step 1 | configure | |
| Step 2 | router hsrp Example: <pre>RP/0/RP0/CPU0:router(config)# router hsrp</pre> | Enables HSRP configuration mode. |

| | Command or Action | Purpose |
|--------|---|---|
| Step 3 | interface <i>type interface-path-id</i> Example: <pre>RP/0/RP0/CPU0:router(config-hsrp)# interface TenGigE 0/2/0/1</pre> | Enables HSRP interface configuration mode on a specific interface. |
| Step 4 | hsrp delay [<i>minimum seconds</i>] [<i>reload seconds</i>] Example: <pre>RP/0/RP0/CPU0:router(config-hsrp-if)#hsrp delay minimum 2 reload 10</pre> | Delays the startup of the state machine when an interface comes up, so that the network has time to settle and there are no unnecessary state changes early after the link comes up. The reload delay is the delay applied after the first interface up event. The minimum delay is the delay that is applied after any subsequent interface up event (if the interface flaps). |
| Step 5 | address-family ipv4 Example: <pre>RP/0/RP0/CPU0:router(config-hsrp-if)# address-family ipv4</pre> | Enables HSRP address-family configuration mode on a specific interface. |
| Step 6 | hsrp group-number version version-no Example: <pre>RP/0/RP0/CPU0:router(config-hsrp-ipv4)# hsrp 1 version 1</pre> | Enables HSRP group submodule. Note The version keyword is available only if IPv4 address-family is selected. By default, version is set to 2 for IPv6 address families. |
| Step 7 | address { learn address [secondary] } Example: <pre>RP/0/RP0/CPU0:router(config-hsrp-gp)# address learn</pre> | Activates HSRP on the configured interface. <ul style="list-style-type: none"> • 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 virtual address is learned from the active router. Note If you configure HSRP for IPv6, you must configure a link local IPv6 address or enable it using the autoconfig keyword. If you do not configure a linklocal IPv6 address, the router does not accept the configuration when you commit your changes using the commit keyword. |
| Step 8 | commit | |

Enabling HSRP Support for ICMP Redirect Messages

By default, HSRP filtering of ICMP redirect messages is enabled on routers running HSRP.

To configure the reenabling of this feature on your router if it is disabled, use the **hsrp redirects** command in interface configuration mode.

SUMMARY STEPS

1. **configure**
2. **router hsrp**
3. **interface** *type interface-path-id*
4. **hsrp redirects disable**
5. **address-family ipv4**
6. **hsrp group-number version** *version-no*
7. **address** { **learn** | *address [secondary]* }
8. **commit**

DETAILED STEPS

| | Command or Action | Purpose |
|---------------|---|---|
| Step 1 | configure | |
| Step 2 | router hsrp Example: RP/0/RP0/CPU0:router(config)# router hsrp | Enables HSRP configuration mode. |
| Step 3 | interface <i>type interface-path-id</i> Example: RP/0/RP0/CPU0:router(config-hsrp)# interface TenGigE 0/2/0/1 | Enables HSRP interface configuration mode on a specific interface. |
| Step 4 | hsrp redirects disable Example: RP/0/RP0/CPU0:router(config-hsrp-if)# hsrp redirects | Configures Internet Control Message Protocol (ICMP) redirect messages to be sent when the Hot Standby Router Protocol (HSRP) is configured on an interface. <ul style="list-style-type: none"> • The hsrp redirects command can be configured on a per-interface basis. When HSRP is first configured on an interface, the setting for that interface inherits the global value. If ICMP redirects have been explicitly disabled on an interface, then the global command cannot reenables the functionality. • With the hsrp redirects command enabled, ICMP redirect messages are filtered by replacing the real IP address in the next-hop address of the redirect packet with a virtual IP address, if it is known to HSRP. • To revert to the default, which is that ICMP messages are enabled, use the no hsrp redirects command. |
| Step 5 | address-family ipv4 Example: RP/0/RP0/CPU0:router(config-hsrp-if)# address-family ipv4 | Enables HSRP address-family configuration mode on a specific interface. |

| | Command or Action | Purpose |
|--------|--|---|
| Step 6 | hsrp <i>group-number</i> version <i>version-no</i> Example: <pre>RP/0/RP0/CPU0:router(config-hsrp-ipv4)# hsrp 1 version 1</pre> | Enables HSRP group submode. Note The version keyword is available only if IPv4 address-family is selected. By default, version is set to 2 for IPv6 address families. |
| Step 7 | address { learn <i>address</i> [secondary] } Example: <pre>RP/0/RP0/CPU0:router(config-hsrp-gp)# address learn</pre> | Activates HSRP on the configured interface. <ul style="list-style-type: none"> 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 virtual address is learned from the active router. Note If you configure HSRP for IPv6, you must configure a link local IPv6 address or enable it using the autoconfig keyword. If you do not configure a linklocal IPv6 address, the router does not accept the configuration when you commit your changes using the commit keyword. |
| Step 8 | commit | |

Multiple Group Optimization (MGO) for HSRP

Multiple Group Optimization provides a solution for reducing control traffic in a deployment consisting of many subinterfaces. By running the HSRP control traffic for just one of the sessions, the control traffic is reduced for the subinterfaces with identical redundancy requirements. All other sessions are subordinates of this primary session, and inherit their states from it.

Customizing HSRP

Customizing the behavior of HSRP is optional. Be aware that as soon as you enable a HSRP group, that group is in operation.

SUMMARY STEPS

- configure**
- router hsrp**
- interface** *type interface-path-id*
- address-family ipv4**
- hsrp** *group-no* **version** *version-no*
- name** *name*
- address** { **learn** | *address* }
- address** *address* **secondary**
- authentication** *string*
- bfd fast-detect**
- mac-address** *address*

12. **hsrp** *group-no slave*
13. **follow** *mgo-session-name*
14. **address** *ip-address*
15. **commit**

DETAILED STEPS

| | Command or Action | Purpose |
|---------------|--|--|
| Step 1 | configure | |
| Step 2 | router hsrp Example: <pre>Router(config)# router hsrp</pre> | Enables HSRP configuration mode. |
| Step 3 | interface <i>type interface-path-id</i> Example: <pre>Router(config-hsrp)# interface TenGigE 0/2/0/1</pre> | Enables HSRP interface configuration mode on a specific interface. |
| Step 4 | address-family ipv4 Example: <pre>Router(config-hsrp-if)# address-family ipv4</pre> | Enables HSRP address-family configuration mode on a specific interface. |
| Step 5 | hsrp <i>group-no version version-no</i> Example: <pre>Router(config-hsrp-ipv4)# hsrp 1 version 2</pre> | Enables HSRP group configuration mode on a specific interface. Note The version keyword is available only if IPv4 address-family is selected. By default, version is set to 2 for IPv6 address families. |
| Step 6 | name <i>name</i> Example: <pre>Router(config-hsrp-gp)# name s1</pre> | Configures an HSRP session name. |
| Step 7 | address { learn <i>address</i> } Example: <pre>Router(config-hsrp-gp)# address learn</pre> | Enables hot standby protocol for IP. • 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 virtual address is learned from the active router. |
| Step 8 | address <i>address secondary</i> Example: <pre>Router(config-hsrp-gp)# address 10.20.30.1</pre> | Configures the secondary virtual IPv4 address for a router. |

| | Command or Action | Purpose |
|----------------|---|---|
| | <code>secondary</code> | |
| Step 9 | authentication <i>string</i> Example: <pre>Router(config-hsrp-gp)# authentication company1</pre> | Configures an authentication string for the Hot Standby Router Protocol (HSRP). |
| Step 10 | bfd fast-detect Example: <pre>Router(config-hsrp-gp)# bfd fast-detect</pre> | Enables bidirectional forwarding(BFD) fast-detection on a HSRP interface. |
| Step 11 | mac-address <i>address</i> Example: <pre>Router(config-hsrp-gp)# mac-address 4000.1000.1060</pre> | Specifies a virtual MAC address for the Hot Standby Router Protocol (HSRP). |
| Step 12 | hsrp <i>group-no slave</i> Example: <pre>Router(config-hsrp-gp)# hsrp 2 slave</pre> | Enables HSRP slave configuration mode on a specific interface. |
| Step 13 | follow <i>mgo-session-name</i> Example: <pre>Router(config-hsrp-slave)# follow s1</pre> | Instructs the subordinate group to inherit its state from a specified group. |
| Step 14 | address <i>ip-address</i> Example: <pre>Router(config-hsrp-slave)# address 10.3.2.2</pre> | Configures the primary virtual IPv4 address for the subordinate group. |
| Step 15 | commit | |

Configuring a Primary Virtual IPv4 Address

To enable hot standby protocol for IP, use the **address (hsrp)** command in the HSRP group submode.

SUMMARY STEPS

1. **configure**
2. **router hsrp**
3. **interface** *type interface-path-id*
4. **address-family ipv4**
5. **hsrp** *group-noversion version-no*
6. **address** { *learn* | *address*}
7. **commit**

DETAILED STEPS

| | Command or Action | Purpose |
|---------------|--|---|
| Step 1 | configure | |
| Step 2 | router hsrp Example: RP/0/RP0/CPU0:router(config)# router hsrp | Enables HSRP configuration mode. |
| Step 3 | interface <i>type interface-path-id</i> Example: RP/0/RP0/CPU0:router(config-hsrp)# interface TenGigE 0/2/0/1 | Enables HSRP interface configuration mode on a specific interface. |
| Step 4 | address-family ipv4 Example: RP/0/RP0/CPU0:router(config-hsrp-if)# address-family ipv4 | Enables HSRP address-family configuration mode on a specific interface. |
| Step 5 | hsrp <i>group-noversion version-no</i> Example: RP/0/RP0/CPU0:router(config-hsrp-ipv4)# hsrp 1 version 2 | Enables HSRP group configuration mode on a specific interface. Note <ul style="list-style-type: none"> • The version keyword is available only if IPv4 address-family is selected. By default, version is set to 2 for IPv6 address families. • HSRP version 2 provides an extended group range of 0-4095. |
| Step 6 | address { <i>learn</i> <i>address</i> } Example: RP/0/RP0/CPU0:router(config-hsrp-ipv4)# address learn | Enables hot standby protocol for IP. |
| Step 7 | commit | |

Configuring a Secondary Virtual IPv4 Address

To configure the secondary virtual IPv4 address for a router, use the **address secondary** command in the Hot Standby Router Protocol (HSRP) virtual router submode.

SUMMARY STEPS

1. **configure**
2. **router hsrp**
3. **interface** *type interface-path-id*
4. **address-family ipv4**
5. **hsrp** *group-noversion version-no*
6. **address** *address* **secondary**
7. **commit**

DETAILED STEPS

| | Command or Action | Purpose |
|--------|---|---|
| Step 1 | configure | |
| Step 2 | router hsrp Example: RP/0/RP0/CPU0:router(config)# router hsrp | Enables HSRP configuration mode. |
| Step 3 | interface <i>type interface-path-id</i> Example: RP/0/RP0/CPU0:router(config-hsrp)# interface TenGigE 0/2/0/1 | Enables HSRP interface configuration mode on a specific interface. |
| Step 4 | address-family ipv4 Example: RP/0/RP0/CPU0:router(config-hsrp-if)# address-family ipv4 | Enables HSRP address-family configuration mode on a specific interface. |
| Step 5 | hsrp <i>group-noversion version-no</i> Example: RP/0/RP0/CPU0:router(config-hsrp-ipv4)# hsrp 1 version 2 | Enables HSRP group configuration mode on a specific interface. Note <ul style="list-style-type: none"> • The version keyword is available only if IPv4 address-family is selected. By default, version is set to 2 for IPv6 address families. • HSRP version 2 provides an extended group range of 0-4095. |
| Step 6 | address <i>address</i> secondary Example: | Configures the secondary virtual IPv4 address for a router. |

| | Command or Action | Purpose |
|---------------|--|---------|
| | RP/0/RP0/CPU0:router(config-hsrp-ipv4)# address 10.20.30.1 secondary | |
| Step 7 | commit | |

Configuring the Subordinate Group to Inherit its State from a Specified Group

To instruct the subordinate group to inherit its state from a specified group, use the following steps:

SUMMARY STEPS

1. **configure**
2. **router hsrp**
3. **interface** *type interface-path-id*
4. **address-family ipv4**
5. **hsrp** *group-no slave*
6. **follow** *mgo-session-name*
7. **commit**

DETAILED STEPS

| | Command or Action | Purpose |
|---------------|---|--|
| Step 1 | configure | |
| Step 2 | router hsrp Example: Router(config)# router hsrp | Enables HSRP configuration mode. |
| Step 3 | interface <i>type interface-path-id</i> Example: Router(config-hsrp)# interface TenGigE 0/2/0/1 | Enables HSRP interface configuration mode on a specific interface. |
| Step 4 | address-family ipv4 Example: Router(config-hsrp-if)# address-family ipv4 | Enables HSRP address-family configuration mode on a specific interface. |
| Step 5 | hsrp <i>group-no slave</i> Example: Router(config-hsrp-ipv4)# hsrp 2 slave | Enables HSRP slave configuration mode on a specific interface. |
| Step 6 | follow <i>mgo-session-name</i> Example: | Instructs the subordinate group to inherit its state from a specified group. |

| | Command or Action | Purpose |
|---------------|---|---------|
| | <code>Router(config-hsrp-slave)# follow m1</code> | |
| Step 7 | <code>commit</code> | |

Configuring a Subordinate Primary Virtual IPv4 Address

To configure the primary virtual IPv4 address for the subordinate group, use the **subordinate primary virtual IPv4 address** command in the HSRP slave submode.

SUMMARY STEPS

1. `configure`
2. `router hsrp`
3. `interface type interface-path-id`
4. `address-family ipv4`
5. `hsrp group-no slave`
6. `address ip-address`
7. `commit`

DETAILED STEPS

| | Command or Action | Purpose |
|---------------|---|---|
| Step 1 | <code>configure</code> | |
| Step 2 | <code>router hsrp</code> Example: <code>Router(config)# router hsrp</code> | Enables HSRP configuration mode. |
| Step 3 | <code>interface type interface-path-id</code> Example: <code>Router(config-hsrp)# interface TenGigE 0/2/0/1</code> | Enables HSRP interface configuration mode on a specific interface. |
| Step 4 | <code>address-family ipv4</code> Example: <code>Router(config-hsrp-if)# address-family ipv4</code> | Enables HSRP address-family configuration mode on a specific interface. |
| Step 5 | <code>hsrp group-no slave</code> Example: <code>Router(config-hsrp-ipv4)# hsrp 2 slave</code> | Enables HSRP slave configuration mode on a specific interface. |
| Step 6 | <code>address ip-address</code> Example: | Configures the primary virtual IPv4 address for the subordinate group. |

| | Command or Action | Purpose |
|---------------|---|---------|
| | Router(config-hsrp-slave)# address 10.2.3.2 | |
| Step 7 | commit | |

Configuring a Secondary Virtual IPv4 address for the Subordinate Group

Perform this task to configure the secondary virtual IPv4 address for the subordinate group.

SUMMARY STEPS

1. **configure**
2. **router hsrp**
3. **interface** *type interface-path-id*
4. **address-family ipv4**
5. **hsrp group-no slave**
6. **address address secondary**
7. **commit**

DETAILED STEPS

| | Command or Action | Purpose |
|---------------|---|---|
| Step 1 | configure | |
| Step 2 | router hsrp Example: Router(config)# router hsrp | Enables HSRP configuration mode. |
| Step 3 | interface <i>type interface-path-id</i> Example: Router(config-hsrp)# interface TenGigE 0/2/0/1 | Enables HSRP interface configuration mode on a specific interface. |
| Step 4 | address-family ipv4 Example: Router(config-hsrp-if)# address-family ipv4 | Enables HSRP address-family configuration mode on a specific interface. |
| Step 5 | hsrp group-no slave Example: Router(config-hsrp-address-family)# hsrp 2 slave | Enables HSRP slave configuration mode on a specific interface. |
| Step 6 | address address secondary Example: | Configures the secondary virtual IPv4 address for a router. |

| | Command or Action | Purpose |
|--------|--|---------|
| | Router(config-hsrp-slave)# address 10.20.30.1 secondary | |
| Step 7 | commit | |

Configuring a Subordinate Virtual MAC Address

To configure the virtual MAC address for the subordinate group, use the **subordinate virtual mac address** command in the HSRP slave submode.

SUMMARY STEPS

1. **configure**
2. **router hsrp**
3. **interface** *type interface-path-id*
4. **address-family ipv4**
5. **hsrp group-no slave**
6. **mac-address address**
7. **commit**

DETAILED STEPS

| | Command or Action | Purpose |
|--------|---|---|
| Step 1 | configure | |
| Step 2 | router hsrp Example: Router(config)# router hsrp | Enables HSRP configuration mode. |
| Step 3 | interface <i>type interface-path-id</i> Example: Router(config-hsrp)# interface TenGigE 0/2/0/1 | Enables HSRP interface configuration mode on a specific interface. |
| Step 4 | address-family ipv4 Example: Router(config-hsrp-if)# address-family ipv4 | Enables HSRP address-family configuration mode on a specific interface. |
| Step 5 | hsrp group-no slave Example: Router(config-hsrp-ipv4)# hsrp 2 slave | Enables HSRP slave configuration mode on a specific interface. |

| | Command or Action | Purpose |
|---------------|---|---|
| Step 6 | mac-address <i>address</i> Example: Router(config-hsrp-slave)# mac-address 10.20.30 | Configures the virtual MAC address for the subordinate group. |
| Step 7 | commit | |

Configuring an HSRP Session Name

To configure an HSRP session name, use the **session name** command in the HSRP group submode.

SUMMARY STEPS

1. **configure**
2. **router hsrp**
3. **interface** *type interface-path-id*
4. **address-family ipv4**
5. **hsrp** *group-noversion version-no*
6. **name** *name*
7. **commit**

DETAILED STEPS

| | Command or Action | Purpose |
|---------------|--|---|
| Step 1 | configure | |
| Step 2 | router hsrp Example: RP/0/RP0/CPU0:router(config)# router hsrp | Enables HSRP configuration mode. |
| Step 3 | interface <i>type interface-path-id</i> Example: RP/0/RP0/CPU0:router(config-hsrp)# interface TenGigE 0/2/0/1 | Enables HSRP interface configuration mode on a specific interface. |
| Step 4 | address-family ipv4 Example: RP/0/RP0/CPU0:router(config-hsrp-if)# address-family ipv4 | Enables HSRP address-family configuration mode on a specific interface. |
| Step 5 | hsrp <i>group-noversion version-no</i> Example: | Enables HSRP group configuration mode on a specific interface. |

| | Command or Action | Purpose |
|---------------|--|--|
| | <pre>RP/0/RP0/CPU0:router(config-hsrp-ipv4)# hsrp 1 version 2</pre> | <p>Note</p> <ul style="list-style-type: none"> • The version keyword is available only if IPv4 address-family is selected. By default, version is set to 2 for IPv6 address families. • HSRP version 2 provides an extended group range of 0-4095. |
| Step 6 | <p>name <i>name</i></p> <p>Example:</p> <pre>RP/0/RP0/CPU0:router(config-hsrp-ipv4)# name s1</pre> | Configures an HSRP session name. |
| Step 7 | commit | |

BFD for HSRP

Bidirectional Forwarding Detection (BFD) is a network protocol used to detect faults between two forwarding engines. BFD sessions can operate in one of the two modes, namely, asynchronous mode or demand mode. In asynchronous mode, both endpoints periodically send hello packets to each other. If a number of those packets are not received, the session is considered down. In demand mode, it is not mandatory to exchange hello packets; either of the hosts can send hello messages, if needed. Cisco supports the BFD asynchronous mode.

Advantages of BFD

- BFD provides failure detection in less than one second.
- BFD supports all types of encapsulation.
- BFD is not tied to any particular routing protocol, supports almost all routing protocols.

BFD Process

HSRP uses BFD to detect link failure and facilitate fast failover times without excessive control packet overhead.

The HSRP process creates BFD sessions as required. When a BFD session goes down, each Standby group monitoring the session transitions to Active state.

HSRP does not participate in any state elections for 10 seconds after a transition to Active state triggered by a BFD session going down.

Configuring BFD

For HSRP, configuration is applied under the existing HSRP-interface sub-mode, with BFD fast failure configurable per HSRP group and the timers (minimum-interface and multiplier) configurable per interface. BFD fast failure detection is disabled by default.

Enabling BFD

SUMMARY STEPS

1. **configure**
2. **router hsrp**
3. **interface** *type interface-path-id*
4. **address-family ipv4**
5. **hsrp** [*group number*] **version** *version-no* **bfd fast-detect** [**peer** **ipv4** *ipv4-address* *interface-type* *interface-path-id*]
6. **commit**

DETAILED STEPS

| | Command or Action | Purpose |
|---------------|--|--|
| Step 1 | configure | |
| Step 2 | router hsrp Example: RP/0/RP0/CPU0:router(config)# router hsrp | Enables HSRP configuration mode. |
| Step 3 | interface <i>type interface-path-id</i> Example: RP/0/RP0/CPU0:router(config-hsrp)# interface TenGigE 0/2/0/1 | Enables HSRP interface configuration mode on a specific interface. |
| Step 4 | address-family ipv4 Example: RP/0/RP0/CPU0:router(config-hsrp-if)# address-family ipv4 | Enables HSRP address-family configuration mode on a specific interface. |
| Step 5 | hsrp [<i>group number</i>] version <i>version-no</i> bfd fast-detect [peer ipv4 <i>ipv4-address</i> <i>interface-type</i> <i>interface-path-id</i>] Example: RP/0/RP0/CPU0:router(config-hsrp-ipv4)# hsrp 1 version 2 bfd fast-detect peer ipv4 10.3.5.2 TenGigE 0/3/4/2 | Enables fast detection on a specific interface. Note <ul style="list-style-type: none"> • The version keyword is available only if IPv4 address-family is selected. By default, version is set to 2 for IPv6 address families. • HSRP version 2 provides an extended group range of 0-4095. |
| Step 6 | commit | |

Modifying BFD timers (minimum interval)

Minimum interval determines the frequency of sending BFD packets to BFD peers (in milliseconds). The default minimum interval is 15ms.

SUMMARY STEPS

1. **configure**
2. **router hsrp**
3. **interface** *type interface-path-id*
4. **hsrp bfd minimum-interval** *interval*
5. **address-family ipv4**
6. **commit**

DETAILED STEPS

| | Command or Action | Purpose |
|--------|---|--|
| Step 1 | configure | |
| Step 2 | router hsrp Example: RP/0/RP0/CPU0:router(config)# router hsrp | Enables HSRP configuration mode. |
| Step 3 | interface <i>type interface-path-id</i> Example: RP/0/RP0/CPU0:router(config-hsrp)# interface TenGigE 0/2/0/1 | Enables HSRP interface configuration mode on a specific interface. |
| Step 4 | hsrp bfd minimum-interval <i>interval</i> Example: RP/0/RP0/CPU0:router(config-hsrp-if)# hsrp bfd minimum-interval 20 | Sets the minimum interval to the specified period. The interval is in milliseconds; range is 15 to 30000 milliseconds. |
| Step 5 | address-family ipv4 Example: RP/0/RP0/CPU0:router(config-hsrp-if)# address-family ipv4 | Enables HSRP address-family configuration mode on a specific interface. |
| Step 6 | commit | |

Modifying BFD timers (multiplier)

Multiplier is the number of consecutive BFD packets which must be missed from a BFD peer before declaring that peer unavailable. The default multiplier is 3.

SUMMARY STEPS

1. **configure**
2. **router hsrp**
3. **interface** *type interface-path-id*

4. **hsrp bfd multiplier** *multiplier*
5. **address-family ipv4**
6. **commit**

DETAILED STEPS

| | Command or Action | Purpose |
|---------------|--|---|
| Step 1 | configure | |
| Step 2 | router hsrp Example: RP/0/RP0/CPU0:router(config)# router hsrp | Enables HSRP configuration mode. |
| Step 3 | interface <i>type interface-path-id</i> Example: RP/0/RP0/CPU0:router(config-hsrp)# interface TenGigE 0/2/0/1 | Enables HSRP interface configuration mode on a specific interface. |
| Step 4 | hsrp bfd multiplier <i>multiplier</i> Example: RP/0/RP0/CPU0:router(config-hsrp-if)# hsrp bfd multiplier 30 | Sets the multiplier to the value. Range is 2 to 50. |
| Step 5 | address-family ipv4 Example: RP/0/RP0/CPU0:router(config-hsrp-if)# address-family ipv4 | Enables HSRP address-family configuration mode on a specific interface. |
| Step 6 | commit | |

Enhanced Object Tracking for HSRP and IP Static

A failure between the active router and the core network cannot be detected using standard HSRP failure detection mechanisms. Object tracking is used to detect such failures. When such a failure occurs, the active router applies a priority decrement to its HSRP session. If this causes its priority to fall below that of the standby router, it will detect this from the HSRP control traffic, and then use this as a trigger to preempt and take over the active role.

Cisco IOS XR software supports up to 512 tracked objects.

The enhanced object tracking for HSRP and IP Static feature provides first-hop redundancy as well as default gateway selection based on IP Service Level Agreement (IPSLA).

See the *Routing Configuration Guide for Cisco CRS Routers*, for more information about enhanced object tracking for static routes.

Configuring object tracking for HSRP

To enable tracking of the named object with the specified decrement, use the following configuration in the HSRP group sub mode.

SUMMARY STEPS

1. **configure**
2. **router hsrp**
3. **interface** type interface-path-id
4. **address-family ipv4**
5. **hsrp** group-number **version** version-no
6. **track** object name [priority-decrement]
7. **commit**

DETAILED STEPS

| | Command or Action | Purpose |
|--------|---|--|
| Step 1 | configure | |
| Step 2 | router hsrp Example: RP/0/RP0/CPU0:router(config)# router hsrp | Enables HSRP configuration mode. |
| Step 3 | interface type interface-path-id Example: RP/0/RP0/CPU0:router(config-hsrp)# interface TenGigE 0/2/0/1 | Enables HSRP interface configuration mode on a specific interface. |
| Step 4 | address-family ipv4 Example: RP/0/RP0/CPU0:router(config-hsrp-if)# address-family ipv4 | Enables HSRP address-family configuration mode on a specific interface. |
| Step 5 | hsrp group-number version version-no Example: RP/0/RP0/CPU0:router(config-hsrp-ipv4)# hsrp 1 version 1 | Enables HSRP group submode. Note The version keyword is available only if IPv4 address-family is selected. By default, version is set to 2 for IPv6 address families. |
| Step 6 | track object name [priority-decrement] Example: RP/0/RP0/CPU0:router(config-hsrp-gp)# track object t1 2 | Enable tracking of the named object with the specified decrement. |
| Step 7 | commit | |

Hot Restartability for HSRP

In the event of failure of a HSRP process in one active group, forced failovers in peer HSRP active router groups should be prevented. Hot restartability supports warm RP failover without incurring forced failovers to peer HSRP routers for active groups.

Configuration Examples for HSRP Implementation on Software

This section provides the following HSRP configuration examples:

Configuring an HSRP Group: Example

The following is an example of enabling HSRP on an interface and configuring HSRP group attributes:

```
configure
router hsrp
interface TenGigE 0/2/0/1
address-family ipv4
hsrp 1
name s1
address 10.0.0.5
timers 100 200
preempt delay 500
priority 20
track TenGigE 0/2/0/2
authentication company0
use-bia
commit
hsrp 2 slave
follow s1
address 10.3.2.2
commit
```

Configuring a Router for Multiple HSRP Groups: Example

The following is an example of configuring a router for multiple HSRP groups:

```
configure
router hsrp
interface TenGigE 0/2/0/3
address family ipv4
hsrp 1
address 1.0.0.5
priority 20
preempt
authentication sclara
hsrp 2
address 1.0.0.6
priority 110
preempt
authentication mtview
hsrp 3
address 1.0.0.7
```

```
preempt
authentication svale
commit
```

Additional References

The following sections provide references related to HSRP

Related Documents

| Related Topic | Document Title |
|---|--|
| QoS commands: complete command syntax, command modes, command history, defaults, usage guidelines, and examples | <i>Quality of Service Commands on Modular QoS Command Reference for Cisco CRS Routers</i> |
| Class-based traffic shaping, traffic policing, low-latency queuing, and Modified Deficit Round Robin (MDRR) | <i>Configuring Modular Quality of Service Congestion Management on Modular QoS Configuration Guide for Cisco CRS Routers</i> |
| WRED, RED, and tail drop | <i>Configuring Modular QoS Congestion Avoidance on Modular QoS Configuration Guide for Cisco CRS Routers</i> |
| HSRP commands | <i>HSRP Commands on IP Addresses and Services Command Reference for Cisco CRS Routers</i> |
| getting started material | <i>Cisco IOS XR Getting Started Guide for the Cisco CRS Router</i> |
| Information about user groups and task IDs | <i>Configuring AAA Services on System Security Configuration Guide for Cisco CRS Routers</i> |

Standards and RFCs

| Standard/RFC | Title |
|---|-------|
| No new or modified standards are supported by this feature, and support for existing standards has not been modified by this feature. | — |

MIBs

| MB | MIBs Link |
|----|--|
| | To locate and download MIBs for selected platforms, Cisco IOS releases, and feature sets, use Cisco MIB Locator found at the following URL: http://www.cisco.com/go/mibs |

Technical Assistance

| Description | Link |
|---|---|
| <p>The Cisco Support website provides extensive online resources, including documentation and tools for troubleshooting and resolving technical issues with Cisco products and technologies.</p> <p>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.</p> <p>Access to most tools on the Cisco Support website requires a Cisco.com user ID and password.</p> | http://www.cisco.com/support |



CHAPTER 11

Implementing LPTS

Local Packet Transport Services (LPTS) maintains tables describing all packet flows destined for the secure domain router (SDR), making sure that packets are delivered to their intended destinations.

For a complete description of the LPTS commands listed in this module, refer to the LPTS Commands module of *IP Addresses and Services Command Reference for Cisco CRS Routers*.

Feature History for Implementing LPTS

| Release | Modification |
|---------------|--|
| Release 3.6.0 | The LPTS policer configuration feature was introduced. |

- [Prerequisites for Implementing LPTS](#) , on page 213
- [Information About Implementing LPTS](#), on page 213
- [Configuring LPTS Policers](#), on page 214
- [Configuration Examples for Implementing LPTS Policers](#), on page 215
- [Additional References](#), on page 220

Prerequisites for Implementing LPTS

The following prerequisites are required to implement LPTS:

You must be in a user group associated with a task group that includes the proper task IDs. The command reference guides include the task IDs required for each command. If you suspect user group assignment is preventing you from using a command, contact your AAA administrator for assistance.

Information About Implementing LPTS

To implement LPTS features mentioned in this document you must understand the following concepts:

LPTS Overview

LPTS uses two components to accomplish this task: the port arbitrator and flow managers. The port arbitrator and flow managers are processes that maintain the tables that describe packet flows for a logical router, known

as the Internal Forwarding Information Base (IFIB). The IFIB is used to route received packets to the correct Route Processor or line card for processing.

LPTS interfaces internally with all applications that receive packets from outside the router. LPTS functions without any need for customer configuration. However, LPTS **show** commands are provided that allow customers to monitor the activity and performance of LPTS flow managers and the port arbitrator.

LPTS Policers

In Cisco IOS XR, the control packets, which are destined to the Route Processor (RP), are policed using a set of ingress policers in the incoming line cards. These policers are programmed statically during bootup by LPTS components. The policers are applied based on the flow type of the incoming control traffic. The flow type is determined by looking at the packet headers. The policer rates for these static ingress policers are defined in a configuration file, which are programmed on the line card during bootup.

You can change the policer values based on the flow types of these set of ingress policers. You are able to configure the rate per policer per node (locally) and globally using the command-line interface (CLI); therefore, overwriting the static policer values.



Note If two different ACLs with same ACEs are applied to an LPTS Policer, only the first ACL applied takes effect. When the first ACL is removed, the second ACL does not take effect on the LPTS Policer. If you want the second ACL to take effect on the LPTS Policer, reconfigure it on the LPTS Policer.

Configuring LPTS Policers

This task allows you to configure the LPTS policers.

SUMMARY STEPS

1. **configure**
2. **lpts pifib hardware police** [location *node-id*]
3. **flow** *flow_type* {rate *rate*}
4. **commit**
5. **show lpts pifib hardware police** [location {all | *node_id*}]

DETAILED STEPS

| | Command or Action | Purpose |
|--------|--|---|
| Step 1 | configure | |
| Step 2 | lpts pifib hardware police [location <i>node-id</i>] Example: RP/0/RP0/CPU0:router(config)# lpts pifib hardware police location 0/2/CPU0 RP/0/RP0/CPU0:router(config-pifib-policer-per-node)# | Configures the ingress policers and enters pifib policer global configuration mode or pifib policer per node configuration mode. The example shows pifib policer per node configuration mode and global. |

| | Command or Action | Purpose |
|---------------|---|---|
| | <pre>RP/0/RP0/CPU0:router(config)# lpts pifib hardware police RP/0/RP0/CPU0:router(config-pifib-policer-global)#</pre> | |
| Step 3 | <p>flow <i>flow_type</i> {rate <i>rate</i>}</p> <p>Example:</p> <pre>RP/0/RP0/CPU0:router(config-pifib-policer-per-node)# flow ospf unicast default rate 20000</pre> | <p>Configures the policer for the LPTS flow type. The example shows how to configure the policer for the ospf flow type.</p> <ul style="list-style-type: none"> • Use the <i>flow_type</i> argument to select the applicable flow type. For information about the flow types, see <i>IP Addresses and Services Command Reference for Cisco CRS Routers</i>. • Use the rate keyword to specify the rate in packets per seconds (PPS). The range is from 0 to 4294967295. <p>Note LPTS policy for ntp-default flow type, supports a flow rate of 100 pps on Cisco ASR 9000 Series Router.</p> <p>Starting with Cisco IOS XR Release 6.1.3, LPTS policy for ntp-default flow type, supports a flow rate higher than 100 pps on Cisco ASR 9000 Series Router.</p> <p>Based on the number of NTP client scale requirement, you can increase the flow rate value to allow higher packets per second (PPS). For example,</p> <pre>lpts pifib hardware police location 0/0/CPU0 flow ntp default rate 1000 flow ntp known rate 1000</pre> |
| Step 4 | commit | |
| Step 5 | <p>show lpts pifib hardware police [location {all <i>node_id</i>}]</p> <p>Example:</p> <pre>RP/0/RP0/CPU0:router# show lpts pifib hardware police location 0/2/cpu0</pre> | <p>Displays the policer configuration value set.</p> <ul style="list-style-type: none"> • (Optional) Use the location keyword to display pre-Internal Forwarding Information Base (IFIB) information for the designated node. The <i>node-id</i> argument is entered in the <i>rack/slot/module</i> notation. • Use the all keyword to specify all locations. |

Configuration Examples for Implementing LPTS Policers

This section provides the following configuration example:

Configuring LPTS Policers: Example

The following example shows how to configure LPTS policers:

```
configure
lpts pifib hardware police
  flow ospf unicast default rate 200
  flow bgp configured rate 200
  flow bgp default rate 100
!
lpts pifib hardware police location 0/2/CPU0
  flow ospf unicast default rate 100
  flow bgp configured rate 300
!
```

The following is the show command and the sample output:

```
show lpts pifib hardware police location 0/2/CPU0
FT - Flow type ID; PPS - Packets per second configured rate
```

| FT | Flow type | Rate (PPS) | Accept/Drop |
|-------|----------------------|------------|-------------|
| 0 | unconfigured-default | 101 | 0/0 |
| 0 | | | |
| | unconfigured-default | | |
| 101 | | | 0/0 |
| 1 | | | |
| | Fragment | | |
| 1000 | | 0 | |
| /0 | | | |
| 2 | | | |
| | OSPF-mc-known | | |
| 1500 | | | |
| 32550 | | | |
| /0 | | | |
| 3 | | | |
| | OSPF-mc | | |
| | -default | | |
| 250 | | | 0/0 |
| 4 | | | |
| | OSPF-uc-known | | |
| 2000 | | | |
| 0 | | | |
| /0 | | | |
| 5 | | | |
| | OSPF | | |
| | -uc-default | | |
| 101 | | | |
| 1 | | | |
| /0 | | | |


```

6
  ISIS-known          250   1500      0/0
7
ISIS
-default

250

0
/0
8
  BGP-known

  2000      17612
/0
9
BGP-default cfg-peer          203

5
/0
10 BGP
-default

500

4
/0
11
PIM-mcast          1500      0/0
12 PIM-ucast       1500      0/0
13 IGMP

      1500
      0/0
14
ICMP-local          1046      0/0
15
ICMP-app           1000      1046      0/0
16
ICMP-control

1000
      0/0
17 ICMP
-default

1046      0
/0
18
LDP-TCP-known      1500      9965
/0
19
LDP-TCP-cfg-peer

1500
0/0
20

```

```

LDP-TCP-default
250
  0
  /0
  21 LDP
  -UDP
1000
59759
/0
22 All
  -routers
      1500      0/0
23
LMP-TCP-known
      1500      0/0
24
LMP-TCP-cfg-peer
1500
0/0
25
LMP-TCP-default
250
      0/0
26 LMP
  -UDP
      1000      0/0
27 RSVP-UDP
      1000      0/0
28 RSVP
1000      0/0
29 IKE
      1000      0/0
30
IPSEC-known
1000
0/0
31 IPSEC
  -default
      250
      0/0
32
MSDP-known
      1000      0/0
33
MSDP-cfg-peer
      1000
0/0
34 MSDP-default
250
      0/0
35 SNMP
1000
0/0
36 NTP

```

```

500
    0/0
37
SSH-known
    1000    0/0
38 SSH
-default
    1000    0/0
39
HTTP-known
    1000    0/0
40 HTTP
-default
1000    0/0
41
SHTTP-known
    1000    0/0
42 SHTTP
-default
    1000    0/0
43
TELNET-known
    500    1000    0/0
44 TELNET
-default

500
    0/0
45
CSS-known

1000
0/0
46 CSS
-default

500
    0/0
47
RSH-known

1000
0/0
48 RSH
-default

500
    0/0
49
UDP-known

2000
    0/0
50
UDP-listen
    1500    0/0
51
UDP-cfg-peer

1500

0
/0
    
```

```

52 UDP
  -default

101
  653
  /0
53
TCP-known
      2000      0/0

54
TCP-listen
      2000      0/0

55
TCP-cfg-peer

2000

0
/0
56 TCP
  -default

101
  6
  /0
57
Mcast-known

2000
0/0
58 Mcast
  -default

101
      0/0
59
Raw-listen
      250      0/0

60 Raw
  -default

250
      0/0
61 ip-sla

1000
      0/0
62 EIGRP
      1500      0/0

63 RIP
      2398      1500      0/0

64
PCEP
      101      0/0

```

Additional References

The following sections provide references related to implementing LPTS.

Related Documents

| Related Topic | Document Title |
|---|---|
| Cisco IOS XR LPTS commands: complete command syntax, command modes, command history, defaults, usage guidelines, and examples | <i>Cisco LPTS Commands</i> module in the <i>IP Addresses and Services Command Reference for Cisco CRS Routers</i> |

Standards

| Standards | Title |
|---|-------|
| No new or modified standards are supported by this feature, and support for existing standards has not been modified by this feature. | — |

MIBs

| MIBs | MIBs Link |
|------|---|
| — | To locate and download MIBs, use the Cisco MIB Locator found at the following URL and choose a platform under the Cisco Access Products menu: https://mibs.cloudapps.cisco.com/ITDIT/MIBS/servlet/index |

RFCs

| RFCs | Title |
|---|-------|
| No new or modified RFCs are supported by this feature, and support for existing RFCs has not been modified by this feature. | — |

Technical Assistance

| Description | Link |
|---|---|
| The Cisco Technical Support website contains thousands of pages of searchable technical content, including links to products, technologies, solutions, technical tips, and tools. Registered Cisco.com users can log in from this page to access even more content. | http://www.cisco.com/techsupport |



CHAPTER 12

Implementing VRRP

The Virtual Router Redundancy Protocol (VRRP) feature allows for transparent failover at the first-hop IP router, enabling a group of routers to form a single virtual router.

Feature History for Implementing VRRP

| Release | Modification |
|---------------|--|
| Release 2.0 | This feature was introduced. |
| Release 3.4.0 | This feature was updated to support the minimum and reload delay options. |
| Release 3.5.0 | VRRP supports Ethernet link bundles. |
| Release 3.7.0 | The clear vrrp statistics command was introduced |
| Release 3.9.0 | <ul style="list-style-type: none">• BFD for VRRP feature was added.• MIB support for VRRP feature was added.• Hot Restartability for VRRP feature was added. |
| Release 4.1.0 | VRRP over IPv6 feature was added. |

- [Prerequisites for Implementing VRRP on Cisco IOS XR Software, on page 224](#)
- [Restrictions for Implementing VRRP on Cisco IOS XR Software , on page 224](#)
- [Information About Implementing VRRP , on page 224](#)
- [Configuring VRRP , on page 227](#)
- [Multiple Group Optimization for Virtual Router Redundancy Protocol, on page 232](#)
- [MIB support for VRRP, on page 237](#)
- [VRRP Support on PWHE Interfaces, on page 237](#)
- [Hot Restartability for VRRP, on page 239](#)
- [Configuration Examples for VRRP Implementation on Cisco IOS XR Software , on page 240](#)
- [Additional References, on page 241](#)

Prerequisites for Implementing VRRP on Cisco IOS XR Software

You must be in a user group associated with a task group that includes the proper task IDs. The command reference guides include the task IDs required for each command. If you suspect user group assignment is preventing you from using a command, contact your AAA administrator for assistance.

Restrictions for Implementing VRRP on Cisco IOS XR Software

The following are restrictions for implementing VRRP:

- Protocol Independent Multicast (PIM) is not supported with VRRP.

Information About Implementing VRRP

To implement VRRP on Cisco IOS XR software, you need to understand the following concepts:

VRRP Overview

A LAN client can use a dynamic process or static configuration to determine which router should be the first hop to a particular remote destination. The client examples of dynamic router discovery are as follows:

- Proxy ARP—The client uses Address Resolution Protocol (ARP) to get the destination it wants to reach, and a router responds to the ARP request with its own MAC address.
- Routing protocol—The client listens to dynamic routing protocol updates (for example, from Routing Information Protocol [RIP]) and forms its own routing table.
- IRDP (ICMP Router Discovery Protocol) client—The client runs an Internet Control Message Protocol (ICMP) router discovery client.

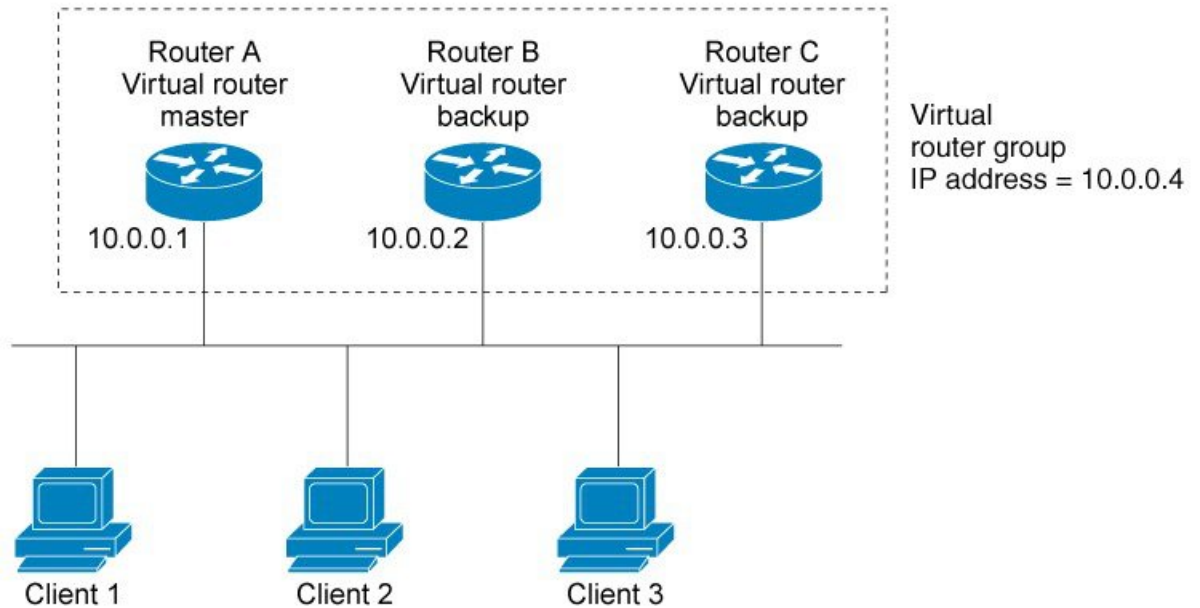
The drawback to dynamic discovery protocols is that they incur some configuration and processing overhead on the LAN client. Also, in the event of a router failure, the process of switching to another router can be slow.

An alternative to dynamic discovery protocols is to statically configure a default router on the client. This approach simplifies client configuration and processing, but creates a single point of failure. If the default gateway fails, the LAN client is limited to communicating only on the local IP network segment and is cut off from the rest of the network.

The Virtual Router Redundancy Protocol (VRRP) feature can solve the static configuration problem. VRRP is an IP routing redundancy protocol designed to allow for transparent failover at the first-hop IP router. VRRP enables a group of routers to form a single *virtual router*. The LAN clients can then be configured with the virtual router as their default gateway. The virtual router, representing a group of routers, is also known as a *VRRP group*.

For example, [Figure 16: Basic VRRP Topology, on page 225](#) shows a LAN topology in which VRRP is configured. In this example, Routers A, B, and C are *VRRP routers* (routers running VRRP) that compose a virtual router. The IP address of the virtual router is the same as that configured for the interface of Router A (10.0.0.1).

Figure 16: Basic VRRP Topology



Because the virtual router uses the IP address of the physical interface of Router A, Router A assumes the role of the *IP address owner*. As the IP address owner router, Router A controls the IP address of the virtual router and is responsible for forwarding packets sent to this IP address. Clients 1 through 3 are configured with the default gateway IP address of 10.0.0.1.

Routers B and C function as *backup virtual routers*. If the IP address owner router fails, the router configured with the higher priority becomes the IP address owner virtual router and provides uninterrupted service for the LAN hosts. When Router A recovers, it becomes the IP address owner virtual router again.



Note We recommend that you disable Spanning Tree Protocol (STP) on switch ports to which the virtual routers are connected. Enable RSTP or rapid-PVST on the switch interfaces if the switch supports these protocols.

Multiple Virtual Router Support

You can configure up to 255 virtual routers on a router physical interface. The actual number of virtual routers that a router interface can support depends on the following factors:

- Router processing capability
- Router memory capability
- Router interface support of multiple MAC addresses

In a topology where multiple virtual routers are configured on a router interface, the interface can act as a IP address owner for one or more virtual routers and as a backup for one or more virtual routers.

VRRP Router Priority

An important aspect of the VRRP redundancy scheme is VRRP router priority. Priority determines the role that each VRRP router plays and what happens if the IP address owner virtual router fails.

If a VRRP router owns the IP address of the virtual router and the IP address of the physical interface, this router functions as a IP address owner virtual router.

Priority also determines if a VRRP router functions as a backup virtual router and determines the order of ascendancy to becoming a IP address owner virtual router if the IP address owner virtual router fails. You can configure the priority of each backup virtual router with a value of 1 through 254, using the **vrrp priority** command.

For example, if Router A, the IP address owner virtual router in a LAN topology, fails, an election process takes place to determine if backup virtual Routers B or C should take over. If Routers B and C are configured with the priorities of 101 and 100, respectively, Router B is elected to become IP address owner virtual router because it has the higher priority. If Routers B and C are both configured with the priority of 100, the backup virtual router with the higher IP address is elected to become the IP address owner virtual router.

By default, a preemptive scheme is enabled whereby a higher-priority backup virtual router that becomes available takes over for the backup virtual router that was elected to become IP address owner virtual router. You can disable this preemptive scheme using the **no vrrp preempt** command. If preemption is disabled, the backup virtual router that is elected to become IP address owner router virtual router remains the IP address owner router until the original IP address owner virtual router recovers and becomes IP address owner router again.

VRRP Advertisements

The IP address owner virtual router sends VRRP advertisements to other VRRP routers in the same group. The advertisements communicate the priority and state of the IP address owner virtual router. The VRRP advertisements are encapsulated in IP packets and sent to the IP Version 4 multicast address assigned to the VRRP group. The advertisements are sent every second by default; the interval is configurable.

Benefits of VRRP

The benefits of VRRP are as follows:

- **Redundancy**—VRRP enables you to configure multiple routers as the default gateway router, 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 routers, thereby sharing the traffic load more equitably among available routers.
- **Multiple Virtual Routers**—VRRP supports up to 255 virtual routers (VRRP groups) on a router physical interface, subject to the platform supporting multiple MAC addresses. Multiple virtual router support enables you to implement redundancy and load sharing in your LAN topology.
- **Multiple IP Addresses**—The virtual router 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.
- **Preemption**—The redundancy scheme of VRRP enables you to preempt a backup virtual router that has taken over for a failing IP address owner virtual router with a higher-priority backup virtual router that has become available.

- Text Authentication—You can ensure that VRRP messages received from VRRP routers that comprise a virtual router are authenticated by configuring a simple text password.
- Advertisement Protocol—VRRP uses a dedicated Internet Assigned Numbers Authority (IANA) standard multicast address (224.0.0.18) for VRRP advertisements. This addressing scheme minimizes the number of routers that must service the multicasts and allows test equipment to accurately identify VRRP packets on a segment. The IANA assigns VRRP the IP protocol number 112.

Configuring VRRP

This section contains instructions for configuring VRRP for IPv4 and IPv6 networks.



Note The VRRP virtual router id (vrid) has to be different for different sub-interfaces, for a given physical interface.

Configuring VRRP for IPv4 Networks

This section describes the procedure for configuring and verifying VRRP for IPv4 networks.

Configuration

Use the following configuration for configuring VRRP for IPv4 networks.



Note Certain customizations (as mentioned) are recommended to control the behavior of the VRRP group on committing the VRRP configuration on the Router. If the following customizations are not configured, then the Router seizes control of the VRRP group, and immediately assumes the role of the IP address owner virtual Router.

```
/* Enter the interface configuration mode and configure an IPv4 address for the interface.
*/
```

```
Router(config)# interface gigabitEthernet 0/0/0/1
Router(config-if)# ipv4 address 10.10.10.1 255.255.255.0
Router(config-if)# no shut
Router(config-if)# commit
Fri Dec  8 13:49:24.142 IST
Router:Dec  8 13:49:24.285 : ifmgr[402]: %PKT_INFRA-LINK-3-UPDOWN : Interface
GigabitEthernet0/0/0/1, changed state to Down
Router:Dec  8 13:49:24.711 : ifmgr[402]: %PKT_INFRA-LINK-3-UPDOWN : Interface
GigabitEthernet0/0/0/1, changed state to Up
```

```
Router(config-if)# exit
Router(config)# do show ip int brief
Fri Dec  8 13:50:05.505 IST
```

| Interface | IP-Address | Status | Protocol | Vrf-Name |
|-------------------------------|-------------------|-----------|-----------|----------------|
| GigabitEthernet0/0/0/0 | unassigned | Shutdown | Down | default |
| GigabitEthernet0/0/0/1 | 10.10.10.1 | Up | Up | default |
| GigabitEthernet0/0/0/2 | unassigned | Shutdown | Down | default |

```
GigabitEthernet0/0/0/3          unassigned      Shutdown        Down            default
GigabitEthernet0/0/0/4          unassigned      Shutdown        Down            default
```

```
/* Enter the VRRP configuration mode and add the configured interface. */
Router(config)# router vrrp
Router(config-vrrp)# interface GigabitEthernet 0/0/0/1

/* CUSTOMIZATION: Configure a delay for the startup of the state machine when the interface
comes up. */
Router(config-vrrp)# delay minimum 2 reload 10 */

/* Configure VRRP version 3 for IPv4 */
Router(config-vrrp-if)# address-family ipv4 vrrp 100 version 3
Router(config-vrrp-virtual-router)# address 10.10.10.1

/* CUSTOMIZATION: Disable the installation of routes for the VRRP virtual addresses. */
Router(config-vrrp-virtual-Router)# accept-mode disable

/* CUSTOMIZATION: Set a priority for the virtual Router. */
Router(config-vrrp-virtual-Router)# priority 254

/* CUSTOMIZATION: Configure a preempt delay value that controls the selection of the IP
address owner virtual Router. */
Router(config-vrrp-virtual-Router)# preempt delay 15

/* CUSTOMIZATION: Configure the interval between successive advertisements by the IP address
owner virtual Router. */
Router(config-vrrp-virtual-Router)#timer 4

/* CUSTOMIZATION: Configure VRRP to track an interface. */
Router(config-vrrp-virtual-Router)# track interface GigabitEthernet0/0/0/1 30

/* Commit the configuration */
Router(config-vrrp-virtual-Router)# commit
```

You have successfully configured VRRP for IPv4 networks.

Validation

Use the following commands to validate the configuration.

```
/* Validate the configuration */
Router(config-vrrp-virtual-router)# do show run interface GigabitEthernet 0/0/0/1
Fri Dec 8 15:04:38.140 IST
interface GigabitEthernet0/0/0/1
  ipv4 address 10.10.10.1 255.255.255.0
!
```

```
Router(config)# show running-config router vrrp
Fri Dec 8 13:50:18.959 IST
router vrrp
  interface GigabitEthernet0/0/0/1
    delay minimum 2 reload 10
    address-family ipv4
    vrrp 100 version 3
    priority 254
    preempt delay 15
```

```

timer 4
track interface GigabitEthernet0/0/0/2 30
address 10.10.10.1
accept-mode disable
!
!
!

```

```

Router(config-vrrp-virtual-router)# do show vrrp ipv4 interface gigabitEthernet 0/0/0/1
Fri Dec 8 15:02:56.952 IST
IPv4 Virtual Routers:
                A indicates IP address owner
                | P indicates configured to preempt
                | |
Interface   vrID Prio A P State   Master addr   VRouter addr
Gi0/0/0/1   100 255 A P Master   local         10.10.10.1

```

```

Router(config-vrrp-virtual-router)# end
Router# show vrrp detail
Fri Dec 8 15:08:36.469 IST
GigabitEthernet0/0/0/1 - IPv4 vrID 100
  State is Master, IP address owner
    1 state changes, last state change 01:19:06
  State change history:
    Dec 8 13:49:30.147 IST Init      -> Master   Delay timer expired
  Last resign sent:      Never
  Last resign received: Never
  Virtual IP address is 10.10.10.1
  Virtual MAC address is 0000.5E00.0164, state is active
  Master router is local
  Version is 3
  Advertise time 1 secs
    Master Down Timer 3.003 (3 x 1 + (1 x 1/256))
  Minimum delay 1 sec, reload delay 5 sec
  Current priority 255
    Configured priority 100, may preempt
    minimum delay 0 secs

```

You have successfully validated VRRP for IPv4 networks.

Configuring VRRP for IPv6 Networks

This section describes the procedure for configuring and verifying VRRP for IPv6 networks.

Configuration

The following sample includes the configuration and customization of VRRP for IPv6 networks.



Note Certain customizations (as mentioned) are recommended to control the behavior of the VRRP group on committing the VRRP configuration on the Router. If the following customizations are not configured, then the Router seizes control of the VRRP group, and immediately assumes the role of the IP address owner virtual Router.

```

/* Enter the interface configuration mode and configure an IPv6 address */
Router# interface GigabitEthernet 0/0/0/2

```

```

Router(config-if)# ipv6 address 10::1/64
Router(config-if)# no shut

/* Exit the interface configuration mode and enter the vrrp configuration mode */
Router(config-if)# exit
Router(config)# Router vrrp

/* Add the configured interface for VRRP */
Router(config-vrrp)# interface GigabitEthernet 0/0/0/2

/* CUSTOMIZATION: Configure a delay for the startup of the state machine when the interface
comes up. */
Router(config-vrrp)# delay minimum 2 reload 10 */

/* Enable the IPv6 global and link local address family on the interface */
Router(config-vrrp-if)# address-family ipv6 vrrp 50
Router(config-vrrp-virtual-Router)# address linklocal autoconfig

/* CUSTOMIZATION: Disable the installation of routes for the VRRP virtual addresses. */
Router(config-vrrp-virtual-Router)# accept-mode disable

/* CUSTOMIZATION: Set a priority for the virtual Router. */
Router(config-vrrp-virtual-Router)# priority 254

/* CUSTOMIZATION: Configure a preempt delay value that controls the selection of the IP
address owner virtual Router. */
Router(config-vrrp-virtual-Router)# preempt delay 15

/* CUSTOMIZATION: Configure the interval between successive advertisements by the IP address
owner virtual Router. */
Router(config-vrrp-virtual-Router)#timer 4

/* CUSTOMIZATION: Configure VRRP to track an interface. */
Router(config-vrrp-virtual-Router)# track interface GigabitEthernet0/0/0/2 30

/* Commit the configuration */
Router(config-vrrp-virtual-Router)# commit

```

You have successfully configured VRRP for IPv6 networks.

Validation

Use the following commands to validate the configuration.

```

/* Validate the configuration */
Router(config-vrrp-virtual-router)# do show run interface GigabitEthernet 0/0/0/2
Fri Dec 8 14:55:48.378 IST
interface GigabitEthernet0/0/0/2
  ipv6 address 10::1/64
  !
-----
Router(config-vrrp-virtual-router)# do show running-config router vrrp
...
router vrrp
  interface GigabitEthernet0/0/0/2
    delay minimum 2 reload 10
    address-family ipv6
      vrrp 50
        priority 254
        preempt delay 15
        timer 4
      track interface GigabitEthernet0/0/0/2 30

```

```

        address linklocal autoconfig
        accept-mode disable
    !
    !
    !

```

```

Router(config-vrrp-virtual-router)# do show vrrp ipv6 interface gigabitEthernet 0/0/0/2
Fri Dec  8 14:59:25.547 IST
IPv6 Virtual Routers:
                A indicates IP address owner
                | P indicates configured to preempt
                | |
Interface   vrID Prio A P State   Master addr   VRouter addr
Gi0/0/0/2      50 254  P Master  local
                                                fe80::200:5eff:fe00:203

```

```

Router(config-vrrp-virtual-router)# end
Router# show vrrp detail
Fri Dec  8 15:08:36.469 IST
GigabitEthernet0/0/0/2 - IPv6 vrID 50
  State is Master
    2 state changes, last state change 00:18:01
  State change history:
    Dec  8 14:50:23.326 IST  Init      -> Backup   Virtual IP configured
    Dec  8 14:50:35.365 IST  Backup   -> Master   Master down timer expired
  Last resign sent:      Never
  Last resign received: Never
Virtual IP address is fe80::200:5eff:fe00:203
Virtual MAC address is 0000.5E00.0203, state is active
Master router is local

  Advertise time 4 secs
    Master Down Timer 12.031 (3 x 4 + (2 x 4/256))
  Minimum delay 2 sec, reload delay 10 sec
  Current priority 254
    Configured priority 254, may preempt
      minimum delay 15 secs
  Tracked items: 1/1 up: 0 decrement
    Object name          State      Decrement
    GigabitEthernet0/0/0/2  Up        30

```

You have successfully validated VRRP for IPv6 networks.

Clearing VRRP Statistics

Use the **clear vrrp statistics** command to clear all the software counters for the specified virtual router.

SUMMARY STEPS

1. **clear vrrp statistics [ipv4 | ipv6] [interfacetype interface-path-id [vrid]]**

DETAILED STEPS

| | Command or Action | Purpose |
|--------|---|--|
| Step 1 | clear vrrp statistics [ipv4 ipv6] [interfacetype interface-path-id [vrid]] | Clears all software counters for the specified virtual router. |

| | Command or Action | Purpose |
|--|--|--|
| | Example: RP/0/RP0/CPU0:router# clear vrrp statistics | <ul style="list-style-type: none"> If no interface is specified, statistics of all virtual routers are removed. |

Disabling State Change Logging

Perform this task to disable the task of logging the VRRP state change events via syslog.

SUMMARY STEPS

1. **configure**
2. **router vrrp**
3. **message state disable**
4. **commit**

DETAILED STEPS

| | Command or Action | Purpose |
|---------------|---|---|
| Step 1 | configure | |
| Step 2 | router vrrp Example: RP/0/RP0/CPU0:router(config)# router vrrp | Enables the VRRP configuration mode. |
| Step 3 | message state disable Example: RP/0/RP0/CPU0:router(config-vrrp)# message state disable RP/0/RP0/CPU0:router(config-vrrp)# | Disables the task of logging the VRRP state change events via syslog. |
| Step 4 | commit | |

Multiple Group Optimization for Virtual Router Redundancy Protocol

Multiple Group Optimization for Virtual Router Redundancy Protocol (VRRP) provides a solution for reducing control traffic in a deployment consisting of many subinterfaces. By running the VRRP control traffic for just one session, the control traffic is reduced for the subinterfaces with identical redundancy requirements. All other sessions are subordinates of this primary session, and inherit their states from it.

Configuring a VRRP Session Name

Perform this task to configure a VRRP session name.

SUMMARY STEPS

1. **configure**
2. **router vrrp**
3. **interface** *type interface-path-id*
4. **address-family ipv4**
5. **vrrp** *group-no*
6. **name** *name*
7. **commit**

DETAILED STEPS

| | Command or Action | Purpose |
|--------|---|---|
| Step 1 | configure | |
| Step 2 | router vrrp Example: RP/0/RP0/CPU0:router(config)# router vrrp | Enables VRRP configuration mode. |
| Step 3 | interface <i>type interface-path-id</i> Example: RP/0/RP0/CPU0:router(config-vrrp)# interface TenGigE 0/2/0/1 | Enables RP interface configuration mode on a specific interface. |
| Step 4 | address-family ipv4 Example: RP/0/RP0/CPU0:router(config-vrrp-if)# address-family ipv4 | Enables VRRP address-family configuration mode on a specific interface. |
| Step 5 | vrrp <i>group-no</i> Example: RP/0/RP0/CPU0:router(config-vrrp-address-family)# vrrp 1 | Enables VRRP group configuration mode on a specific interface. |
| Step 6 | name <i>name</i> Example: RP/0/RP0/CPU0:router(config-vrrp-vritual-router)# name s1 | Configures a VRRP session name. |
| Step 7 | commit | |

Configuring the Subordinate Group to Inherit its State from a Specified Group (VRRP)

Perform this task to instruct the subordinate group to inherit its state from a specified group.

SUMMARY STEPS

1. **configure**
2. **router vrrp**
3. **interface** *type interface-path-id*
4. **address-family ipv4**
5. **vrrp group-no slave**
6. **follow** *mgo-session-name*
7. **commit**

DETAILED STEPS

| | Command or Action | Purpose |
|---------------|---|--|
| Step 1 | configure | |
| Step 2 | router vrrp Example: Router(config)# router vrrp | Enables VRRP configuration mode. |
| Step 3 | interface <i>type interface-path-id</i> Example: Router(config-vrrp)# interface TenGigE 0/2/0/1 | Enables VRRP interface configuration mode on a specific interface. |
| Step 4 | address-family ipv4 Example: Router(config-vrrp-if)# address-family ipv4 | Enables VRRP address-family configuration mode on a specific interface. |
| Step 5 | vrrp group-no slave Example: Router(config-vrrp-address-family)# vrrp 2 slave | Enables VRRP slave configuration mode on a specific interface. |
| Step 6 | follow <i>mgo-session-name</i> Example: Router(config-vrrp-slave)# follow m1 | Instructs the subordinate group to inherit its state from a specified group. |
| Step 7 | commit | |

Configuring a Primary Virtual IPv4 Address for a Subordinate Group(VRRP)

Perform this task to configure the primary virtual IPv4 address for the subordinate group.

SUMMARY STEPS

1. **configure**
2. **router vrrp**
3. **interface** *type interface-path-id*
4. **address-family ipv4**
5. **vrrp group-no slave**
6. **address ip-address**
7. **commit**

DETAILED STEPS

| | Command or Action | Purpose |
|--------|---|---|
| Step 1 | configure | |
| Step 2 | router vrrp Example: Router(config)# router vrrp | Enables VRRP configuration mode. |
| Step 3 | interface <i>type interface-path-id</i> Example: Router(config-vrrp)# interface TenGigE 0/2/0/1 | Enables VRRP interface configuration mode on a specific interface. |
| Step 4 | address-family ipv4 Example: Router(config-hsrp-if)# address-family ipv4 | Enables VRRP address-family configuration mode on a specific interface. |
| Step 5 | vrrp group-no slave Example: Router(config-vrrp-address-family)# vrrp 2 slave | Enables VRRP slave configuration mode on a specific interface. |
| Step 6 | address ip-address Example: Router(config-vrrp-slave)# address 10.2.3.2 | Configures the primary virtual IPv4 address for the subordinate group. |
| Step 7 | commit | |

Configuring a Secondary Virtual IPv4 address for the Subordinate Group

Perform this task to configure the secondary virtual IPv4 address for the subordinate group.

SUMMARY STEPS

1. **configure**
2. **router hsrp**
3. **interface** *type interface-path-id*
4. **address-family ipv4**
5. **hsrp** *group-no* **slave**
6. **address** *address* **secondary**
7. **commit**

DETAILED STEPS

| | Command or Action | Purpose |
|---------------|---|---|
| Step 1 | configure | |
| Step 2 | router hsrp Example: Router(config)# router hsrp | Enables HSRP configuration mode. |
| Step 3 | interface <i>type interface-path-id</i> Example: Router(config-hsrp)# interface TenGigE 0/2/0/1 | Enables HSRP interface configuration mode on a specific interface. |
| Step 4 | address-family ipv4 Example: Router(config-hsrp-if)# address-family ipv4 | Enables HSRP address-family configuration mode on a specific interface. |
| Step 5 | hsrp <i>group-no</i> slave Example: Router(config-hsrp-address-family)# hsrp 2 slave | Enables HSRP slave configuration mode on a specific interface. |
| Step 6 | address <i>address</i> secondary Example: Router(config-hsrp-slave)# address 10.20.30.1 secondary | Configures the secondary virtual IPv4 address for a router. |
| Step 7 | commit | |

MIB support for VRRP

VRRP enables one or more IP addresses to be assumed by a router when a failure occurs. For example, when IP traffic from a host reaches a failed router because the failed router is the default gateway, the traffic is transparently forwarded by the VRRP router that has assumed control. VRRP does not require configuration of dynamic routing or router discovery protocols on every end host. The VRRP router controlling the IP address(es) associated with a virtual router is called the IP address owner router, and forwards packets sent to these IP addresses. The election process provides dynamic fail over(standby) in the forwarding responsibility should the IP address owner router become unavailable. This allows any of the virtual router IP addresses on the LAN to be used as the default first hop router by end-hosts. The advantage gained from using VRRP is a higher availability default path without requiring configuration of dynamic routing or router discovery protocols on every end-host. SNMP traps provide information of the state changes, when the virtual routers(in standby) are moved to IP address owner router's state or if the standby router is made IP address owner router.

Configuring SNMP server notifications for VRRP events

The `snmp-server traps vrrp events` command enables the Simple Network Management Protocol (SNMP) server notifications (traps) for VRRP.

SUMMARY STEPS

1. `configure`
2. `snmp-server traps vrrp events`
3. `commit`

DETAILED STEPS

| | Command or Action | Purpose |
|--------|--|---|
| Step 1 | <code>configure</code> | |
| Step 2 | <code>snmp-server traps vrrp events</code> Example: <pre>RP/0/RP0/CPU0:router(config)#snmp-server traps vrrp events</pre> | Enables the SNMP server notifications for VRRP. |
| Step 3 | <code>commit</code> | |

VRRP Support on PWHE Interfaces

Pseudowire Headend (PWHE) is a technology that allows termination of access pseudowires (PWs) into a Layer 3 (VRF or global) domain or into a Layer 2 domain. This feature enables you to configure VRRP on PWHE interfaces to provide redundancy between two routers that are connected through PWHE interfaces .

For more information about PWHE interfaces, see the chapter *Implementing Multipoint Layer 2 Services of the L2VPN and Ethernet Services Configuration Guide for Cisco ASR 9000 Series Routers*.

Configuration Example

To configure VRRP on PWHE interfaces, use the following steps:

1. Enter the VRRP configuration mode.
2. Configure a PWHE interface.
3. Configure the VRRP address family for IPv4 and IPv6.

Configuration

```
/* Enter the VRRP configuration mode. */
Router# configure
Router(config)# router vrrp

/* Configure a PWHE interface. */
Router# (config-vrrp)# interface pw-Ether 1000

/* Configure the VRRP address family for IPv4 and IPv6. */
Router(config-vrrp-if)# address-family ipv4 vrrp
Router(config-vrrp-virtual-router)# address 172.16.0.0
Router(config-vrrp-virtual-router)# vrrp 1
Router(config-vrrp-virtual-router)# commit
Router(config-vrrp-address-family)# exit
Router(config-vrrp-if)# exit
Router(config-vrrp-if)# address-family ipv6 vrrp 1
Router(config-vrrp-virtual-router)# address global 2001:DB8::1
Router(config-vrrp-virtual-router)# address linklocal autoconfig
Router(config-vrrp-virtual-router)# commit
```

Running Configuration

```
router vrrp
 interface PW-Ether1000
   address-family ipv4
     vrrp 1
     address 172.16.0.0
   !
 !
 address-family ipv6
   vrrp 1
   address global 2001:db8::1
   address linklocal autoconfig
 !
```

Verification

Use the following command to verify the configuration of VRRP on PWHE interfaces:

```
Router# show run interface pw-ether 1000
interface PW-Ether1000
  ipv4 address 172.16.0.0 255.255.255.0
  ipv6 address 2001:DB8::1/125
  attach generic-interface-list pwhe_vrrp
 !
```

Use the following command to verify the details of VRRP configuration on PWHE interfaces:

```
Router# show vrrp interface pw-Ether 1000 detail
PW-Ether1000 - IPv4 vrID 1
  State is Backup
```

```

    1 state changes, last state change 2d08h
    State change history:
    Nov 24 11:47:16.585 IST Init
    Last resign sent:      Never
    Last resign received: Never
    Virtual IP address is 172.16.0.0
    Virtual MAC address is 0000.5E00.0101, state is reserved
    Master router is 172.16.0.1, priority 100
    Version is 2
    Advertise time 1 secs
    Master Down Timer 3.609 (3 x 1 + (156 x 1/256))
    Minimum delay 1 sec, reload delay 5 sec
    Current priority 100
    Configured priority 100, may preempt
    minimum delay 0 secs

```

PW-Ether1000 - IPv6 vrID 1

```

    State is Backup
    1 state changes, last state change 2d08h
    State change history:
    Nov 24 11:47:19.600 IST Init
    Last resign sent:      Never
    Last resign received: Never
    Virtual IP address is 2001:DB8::1/125
    Secondary Virtual IP address is 2001:DB8:FFFF:FFFF:FFFF:FFFF:FFFF:FFFF
    Virtual MAC address is 0000.5E00.0201, state is reserved
    Master router is 2001:DB8::2
    Version is 3
    Advertise time 1 secs
    Master Down Timer 3.609 (3 x 1 + (156 x 1/256))
    Minimum delay 1 sec, reload delay 5 sec
    Current priority 100
    Configured priority 100, may preempt
    minimum delay 0 secs

```

Use the following command to verify VRRP state and priority of the current router:

```

Router# show vrrp interface pw-Ether 1000
IPV4 Virtual Routers:
                A indicates IP address owner
                | P indicates configured to preempt
                | |
Interface   vrID Prio A P State   Master addr   VRouter addr
PE1000      1 100 P Backup 172.16.0.1   172.16.0.0
IPV6 Virtual Routers:
                A indicates IP address owner
                | P indicates configured to preempt
                | |
Interface   vrID Prio A P State   Master addr   VRouter addr
PE1000      1 100 P Backup 2001:DB8::2   fe80::200:5eff:fe00:201

```

Hot Restartability for VRRP

In the event of failure of a VRRP process in one group, forced failovers in peer VRRP IP address owner router groups should be prevented. Hot restartability supports warm RP failover without incurring forced failovers to peer VRRP routers.

Configuration Examples for VRRP Implementation on Cisco IOS XR Software

This section provides the following VRRP configuration examples:

Configuring a VRRP Group: Example

This section provides the following configuration example of Router A and Router B, each belonging to three VRRP groups:

Router A:

```
config
interface tenGigE 0/4/0/4
ipv4 address 10.1.0.1/24
exit
router vrrp
interface tenGigE 0/4/0/4
address-family ipv4
vrrp 1 version 2
priority 120
text-authentication cisco
timer 3
address 10.0.0.100
vrrp 5 version 2
timer 30
address 10.0.0.105
vrrp 5 version 2
preempt disable
address 10.0.0.200
commit
```

Router B:

```
config
interface tenGigE 0/4/0/4
ipv4 address 10.1.0.2/24
exit
router vrrp
interface tenGigE 0/4/0/4
address-family ipv4
vrrp 1 version 2
priority 100
text-authentication cisco
timer 3
address 10.0.0.100
vrrp 5 version 2
priority 200
timer 30
address 10.0.0.105
vrrp 5 version 2
preempt disable
address 10.0.0.200
```



```
commit
```

In the configuration example, each group has the following properties:

- Virtual Router 1:
 - Virtual IP address is 10. 0.0. 100.
 - Router A will become the IP address owner router for this group with priority 120.
 - Advertising interval is 3 seconds.
 - Preemption is enabled.
 - Authentication is enabled.

- Virtual Router 5:
 - Virtual IP address is 10.0.0.105.
 - Whichever router comes up first will become IP address owner router (as preemption is disabled).
 - Advertising interval is 30 seconds.
 - Preemption is disabled.
 - Authentication is disabled.

- Virtual Router 100:
 - Virtual IP address is 10.0.0.200.
 - Router B will become IP address owner router for this group first, because it has a higher interface IP address (10.0.0.2).
 - Advertising interval is the default 1 second.
 - Preemption is enabled.
 - Authentication is disabled.

Clearing VRRP Statistics: Example

The **clear vrrp statistics** command produces no output of its own. The command modifies the statistics given by **show vrrp statistics** command so that all the statistics are reset to zero.

The following section provides examples of the output of the **show vrrp statistics** command followed by the **clear vrrp statistics** command:

Additional References

The following sections provide references related to VRRP.

Related Documents

| Related Topic | Document Title |
|---|--|
| QoS commands: complete command syntax, command modes, command history, defaults, usage guidelines, and examples | <i>Quality of Service Commands on Modular QoS Command Reference for Cisco CRS Routers</i> |
| Class-based traffic shaping, traffic policing, low-latency queuing, and Modified Deficit Round Robin (MDRR) | <i>Configuring Modular Quality of Service Congestion Management on Modular QoS Configuration Guide for Cisco CRS Routers</i> |
| WRED, RED, and tail drop | <i>Configuring Modular QoS Congestion Avoidance on Modular QoS Configuration Guide for Cisco CRS Routers</i> |
| VRRP commands | <i>VRRP Commands on IP Addresses and Services Command Reference for Cisco CRS Routers</i> |
| getting started material | <i>Cisco IOS XR Getting Started Guide for the Cisco CRS Router</i> |
| Information about user groups and task IDs | <i>Configuring AAA Services on System Security Configuration Guide for Cisco CRS Routers</i> |

Standards

| Standards | Title |
|---|-------|
| No new or modified standards are supported by this feature, and support for existing standards has not been modified by this feature. | — |

MIBs

| MIBs | MIBs Link |
|------|---|
| — | To locate and download MIBs, use the Cisco MIB Locator found at the following URL and choose a platform under the Cisco Access Products menu: https://mibs.cloudapps.cisco.com/ITDIT/MIBS/servlet/index |

RFCs

| RFCs | Title |
|---|-------|
| No new or modified RFCs are supported by this feature, and support for existing RFCs has not been modified by this feature. | — |

Technical Assistance

| Description | Link |
|---|---|
| The Cisco Technical Support website contains thousands of pages of searchable technical content, including links to products, technologies, solutions, technical tips, and tools. Registered Cisco.com users can log in from this page to access even more content. | http://www.cisco.com/techsupport |



CHAPTER 13

Configuring Transports

This module provides information about Nonstop Routing (NSR), Stream Control Transmission Protocol (SCTP), Transmission Control Protocol (TCP), User Datagram Protocol (UDP), and RAW Transports.

If you have specific requirements and need to adjust the NSR, SCTP, TCP, UDP, or RAW values, refer to the *Transport Stack Commands on IP Addresses and Services Command Reference for Cisco CRS Routers*.

Feature History for Configuring NSR, SCTP, TCP, UDP, and UDP RAW Transports on the Cisco IOS XR Software

| Release | Modification |
|---------------|---|
| Release 3.6.0 | The following features were introduced: <ul style="list-style-type: none">• Nonstop Routing (NSR)• Stream Control Transmission Protocol (SCTP) |

- [Prerequisites for Configuring NSR, SCTP, TCP, UDP, and RAW Transports, on page 245](#)
- [Information About Configuring NSR, SCTP, TCP, UDP, and RAW Transports, on page 246](#)
- [How to Configure Failover as a Recovery Action for NSR, on page 247](#)
- [Additional References, on page 248](#)

Prerequisites for Configuring NSR, SCTP, TCP, UDP, and RAW Transports

The following prerequisites are required to implement NSR, SCTP, TCP, UDP, and RAW Transports:

You must be in a user group associated with a task group that includes the proper task IDs. The command reference guides include the task IDs required for each command. If you suspect user group assignment is preventing you from using a command, contact your AAA administrator for assistance.

Information About Configuring NSR, SCTP, TCP, UDP, and RAW Transports

To configure NSR, SCTP, TCP, UDP, and RAW transports, you must understand the following concepts:

NSR Overview

Nonstop Routing (NSR) is provided for Open Shortest Path First (OSPF) and Label Distribution Protocol (LDP) protocols for the following events:

- Route Processor (RP) or Distributed Route Processor (DRP) failover
- Process restart for either OSPF, LDP, or TCP
- Minimum Disruption Restart (MDR)
- Rack online insertion and removal (OIR) in a Cisco CRS-1 Multishelf System

In the case of the RP failover, NSR is achieved by for both TCP and the applications (OSPF or LDP).

NSR is a method to achieve High Availability (HA) of the routing protocols. TCP connections and the routing protocol sessions are migrated from the active RP to standby RP after the RP failover without letting the peers know about the failover. Currently, the sessions terminate and the protocols running on the standby RP reestablish the sessions after the standby RP goes active. Graceful Restart (GR) extensions are used in place of NSR to prevent traffic loss during an RP failover but GR has several drawbacks.

You can use the **nsr process-failures switchover** command to let the RP failover be used as a recovery action when the active TCP or active LDP restarts. When standby TCP or LDP restarts, only the NSR capability is lost till the standby instances come up and the sessions are resynchronized but the sessions do not go down. In the case of the process failure of an active OSPF, a fault-management policy is used. For more information, refer to *Implementing OSPF on Routing Configuration Guide for Cisco CRS Routers*.

SCTP Overview

Stream Control Transmission Protocol (SCTP) is a reliable transport protocol that provides multihoming, stream support, and partial reliability. Multihoming occurs when one (or both) endpoints of a connection can consist of more than one IP address, which enables transparent failover between redundant network paths. SCTP can transport multiple message-streams.

TCP Overview

TCP is a connection-oriented protocol that specifies the format of data and acknowledgments that two computer systems exchange to transfer data. TCP also specifies the procedures the computers use to ensure that the data arrives correctly. TCP allows multiple applications on a system to communicate concurrently, because it handles all demultiplexing of the incoming traffic among the application programs.

UDP Overview

The User Datagram Protocol (UDP) is a connectionless transport-layer protocol that belongs to the IP family. UDP is the transport protocol for several well-known application-layer protocols, including Network File System (NFS), Simple Network Management Protocol (SNMP), Domain Name System (DNS), and TFTP.

Any IP protocol other than TCP, UDP, or SCTP is known as a RAW protocol.

For most sites, the default settings for the TCP, UDP, and RAW transports need not be changed.

How to Configure Failover as a Recovery Action for NSR

This section contains the following procedure:

Configuring Failover as a Recovery Action for NSR

This task allows you to configure failover as a recovery action to process failures of active instances.

When the active TCP or the NSR client of the active TCP terminates or restarts, the TCP sessions go down. To continue to provide NSR, failover is configured as a recovery action. If failover is configured, a switchover is initiated if the active TCP or an active application (for example, LDP, OSPF, and so forth) restarts or terminates.

For information on how to configure MPLS Label Distribution Protocol (LDP) for NSR, refer to the *MPLS Configuration Guide for the Cisco CRS Routers*.

For information on how to configure NSR on a per-process level for each Open Shortest Path First (OSPF) process, refer to the *Routing Configuration Guide for Cisco CRS Routers*.



Note Before performing this procedure, enable RP isolation using the **isolation enable** command for improved troubleshooting. Without enabling RP isolation, the failing process will not generate the logs required to find the root cause of the failure.

SUMMARY STEPS

1. **configure**
2. nsr process-failures switchover
3. **commit**

DETAILED STEPS

| | Command or Action | Purpose |
|--------|--|--|
| Step 1 | configure | |
| Step 2 | nsr process-failures switchover Example: RP/0/RP0/CPU0:router(config)# nsr process-failures | Configures failover as a recovery action for active instances to switch over to a standby route processor (RP) or a distributed route processor (DRP) to maintain nonstop routing (NSR). |

| | Command or Action | Purpose |
|---------------|-------------------|---------|
| | switchover | |
| Step 3 | commit | |

Additional References

The following sections provide references related to configuring NSR, SCTP, TCP, UDP, and RAW transports.

Related Documents

| Related Topic | Document Title |
|---|--|
| the Cisco IOS XR Software Transport Stack commands: complete command syntax, command modes, command history, defaults, usage guidelines, and examples | <i>Transport Stack Commands in the IP Addresses and Services Command Reference for Cisco CRS Routers</i> |
| the Cisco IOS XR Software MPLS LDP commands: complete command syntax, command modes, command history, defaults, usage guidelines, and examples | <i>MPLS Label Distribution Protocol Commands in the MPLS Command Reference for the Cisco CRS Router</i> |
| the Cisco IOS XR Software OSPF commands: complete command syntax, command modes, command history, defaults, usage guidelines, and examples | <i>OSPF Commands in the Routing Command Reference for Cisco CRS Routers</i> |
| MPLS Label Distribution Protocol feature information | <i>Implementing MPLS Label Distribution Protocol in the MPLS Configuration Guide for the Cisco CRS Routers</i> |
| OSPF feature information | <i>Implementing OSPF in the Routing Configuration Guide for Cisco CRS Routers</i> |

Standards

| Standards | Title |
|---|-------|
| No new or modified standards are supported by this feature, and support for existing standards has not been modified by this feature. | — |

MIBs

| MIBs | MIBs Link |
|------|---|
| — | To locate and download MIBs, use the Cisco MIB Locator found at the following URL and choose a platform under the Cisco Access Products menu: https://mibs.cloudapps.cisco.com/ITDIT/MIBS/servlet/index |

RFCs

| RFCs | Title |
|---|-------|
| No new or modified RFCs are supported by this feature, and support for existing RFCs has not been modified by this feature. | — |

Technical Assistance

| Description | Link |
|---|---|
| The Cisco Technical Support website contains thousands of pages of searchable technical content, including links to products, technologies, solutions, technical tips, and tools. Registered Cisco.com users can log in from this page to access even more content. | http://www.cisco.com/techsupport |

