IP Addresses and Services Configuration Guide for Cisco NCS 6000 Routers, IOS XR Release 6.3.x

First Published: 2017-09-15

Last Modified: 2018-03-01

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

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.

- 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.

CONTENTS

PREFACE

CHAPTER 1

CHAPTER 2

CHAPTER 4

CHAPTER 5

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 NCS 6000 Series Routers* preface contains these sections:

Changes to This Document

This table lists the technical changes made to this document since it was first released.

Table 1: Changes to This Document

Date	Change Summary
August 2018	Republished for Release 6.3.3.
March 2018	Republished for Release 6.3.2.
September 2017	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 NCS 6000 Series Routers*, and tells you where they are documented.

New and Changed IP Addresses and Services Features

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

IP Addresses Features Added or Modified in IOS XR Release 6.3.x

Table 2: New and Changed Features

Feature	Description	Introduced/Changed in Release	Where Documented
No new features were introduced.	NA	Release 6.3.1	NA

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 the Cisco NCS Router running Cisco IOS-XR software.

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 NCS 6000 Series Routers*.

Feature History for Implementing Network Stack IPv4 and IPv6

Release	Modification
Release 5.0.0	This feature was introduced.

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 HundredGigE 0/1/0/1 conflicts with HundredGigE 0/2/0/1, then the IPv4 protocol on HundredGigE 0/2/0/1 is brought down and IPv4 remains active on HundredGigE 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, I*nternet 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:

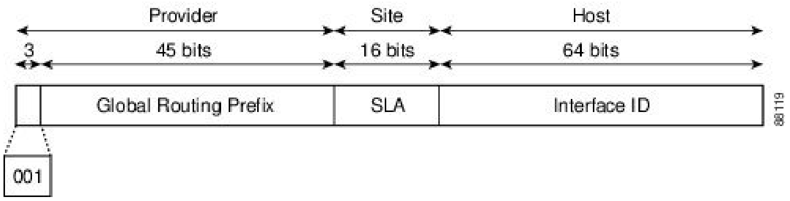
- Global aggregatable address
- Site-local address (proposal to remove by IETF)
- Link-local address
- IPv4-compatible IPv6 address

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 7shows 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 TLS 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 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:

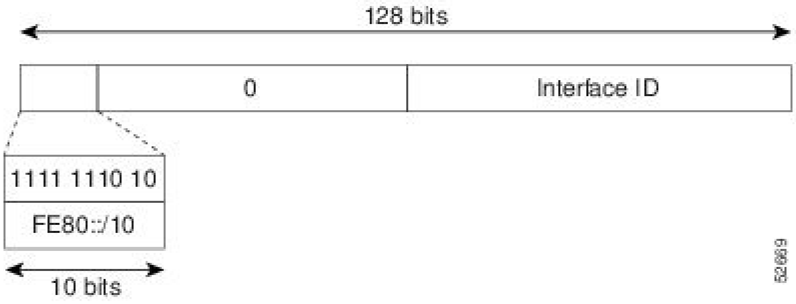
1. The router is queried for MAC addresses (from the pool of MAC addresses in the router).
2. If no MAC address is available, the serial number of the Route Processor (RP) or line card (LC) is used to form the link-local address.

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 8shows the structure of a link-local address.

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

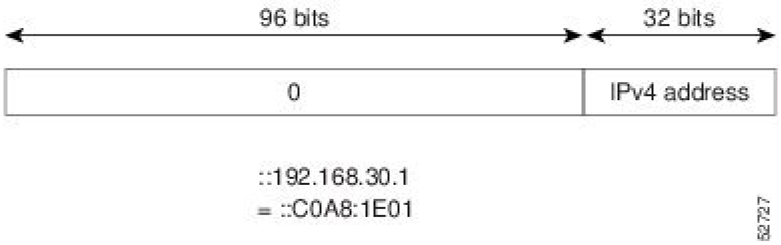
Figure 2: Link-Local Address Format



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 0:0:0:0:0:0: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.

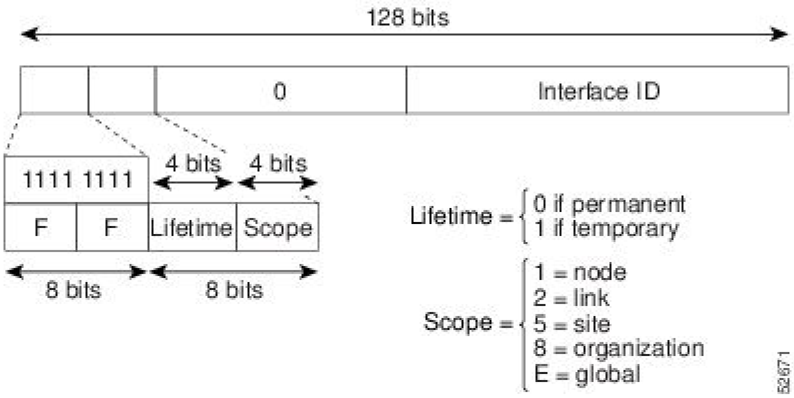
Figure 3: IPv4-Compatible IPv6 Address Format



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



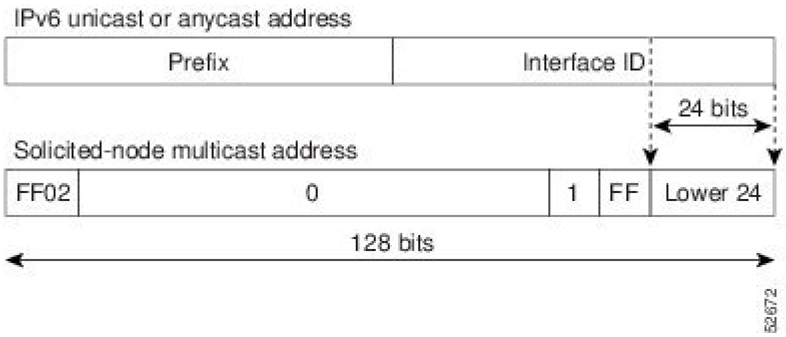
IPv6 nodes (hosts and routers) are required to join (receive packets destined for) the following multicast groups:

- 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


IPv6 unicast or anycast address
Prefix
Interface ID
24 bits
Solicited-node multicast address
FF02
0
1
FF
Lower 24
128 bits
52672

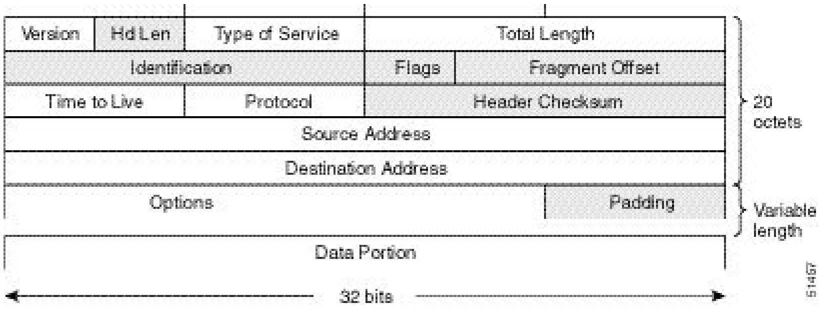
Note There are no broadcast addresses in IPv6. IPv6 multicast addresses are used instead of broadcast addresses.

For further information on IPv6 multicast, refer to the *Implementing Multicast* module in the *Multicast Configuration Guide for the Cisco NCS 6000 Series Routers* .

Simplified IPv6 Packet Header

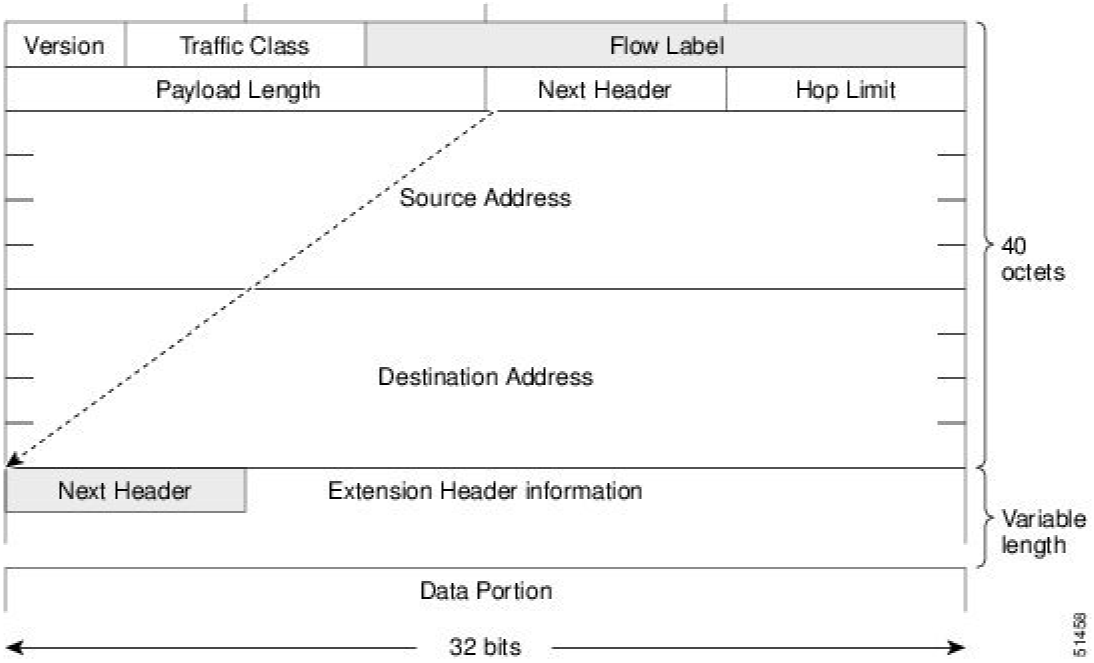
The basic IPv4 packet header has 12 fields with a total size of 20 octets (160 bits). The 12 fields may be followed by an Options field, which is followed by a data portion that is usually the transport-layer packet. The variable length of the Options field adds to the total size of the IPv4 packet header. 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.

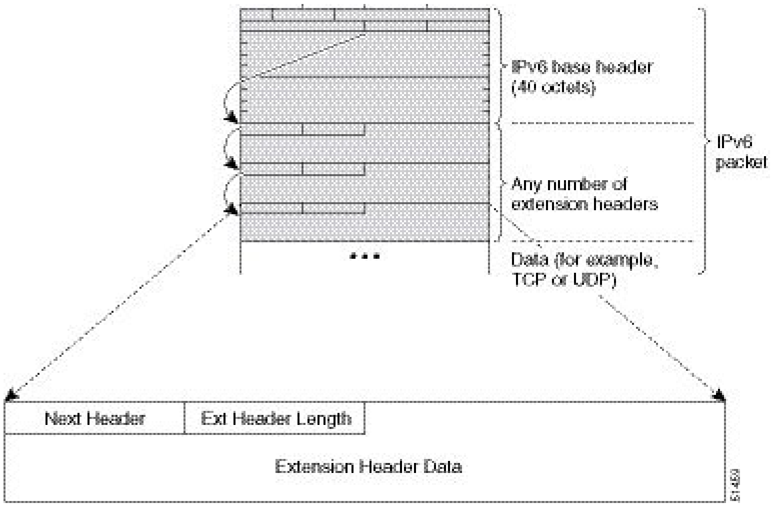
Table 4: Basic IPv6 Packet Header Fields

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.
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.

Field	Description
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 12shows 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.
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.

Header Type	Next Header Value	Description
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.

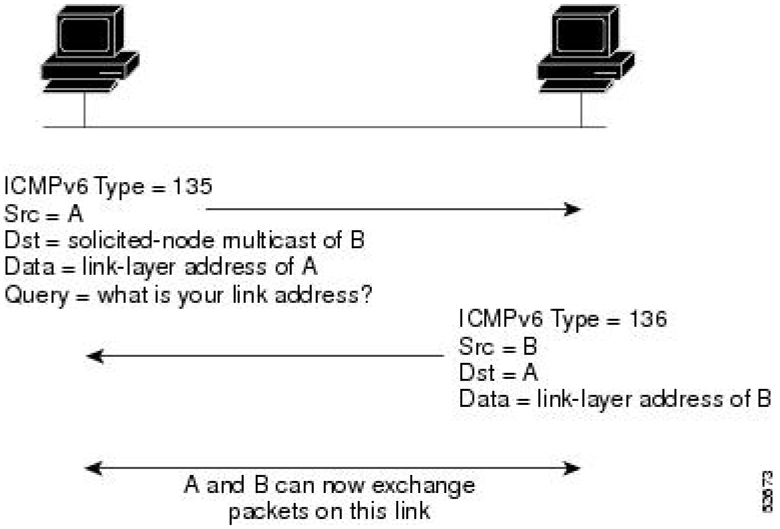
IPv6 Neighbor Discovery

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


ICMPv6 Type = 135
Src = A
Dst = solicited-node multicast of B
Data = link-layer address of A
Query = what is your link address?
ICMPv6 Type = 136
Src = B
Dst = A
Data = link-layer address of B
A and B can now exchange packets on this link

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 verifying 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.

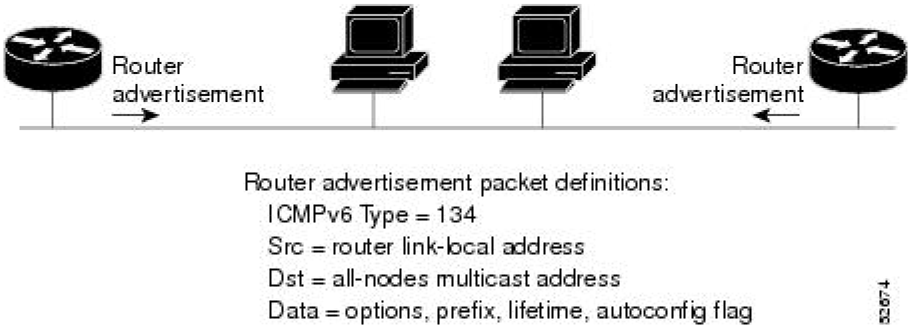
Neighbor solicitation messages are also used in the stateless autoconfiguration process to verify the uniqueness of unicast IPv6 addresses before the addresses are assigned to an interface. Duplicate address detection is performed first on a new, link-local IPv6 address before the address is assigned 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
Router advertisement
Router advertisement packet definitions:
ICMPv6 Type = 134
Src = router link-local address
Dst = all-nodes multicast address
Data = options, prefix, lifetime, autoconfig flag
52674

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)

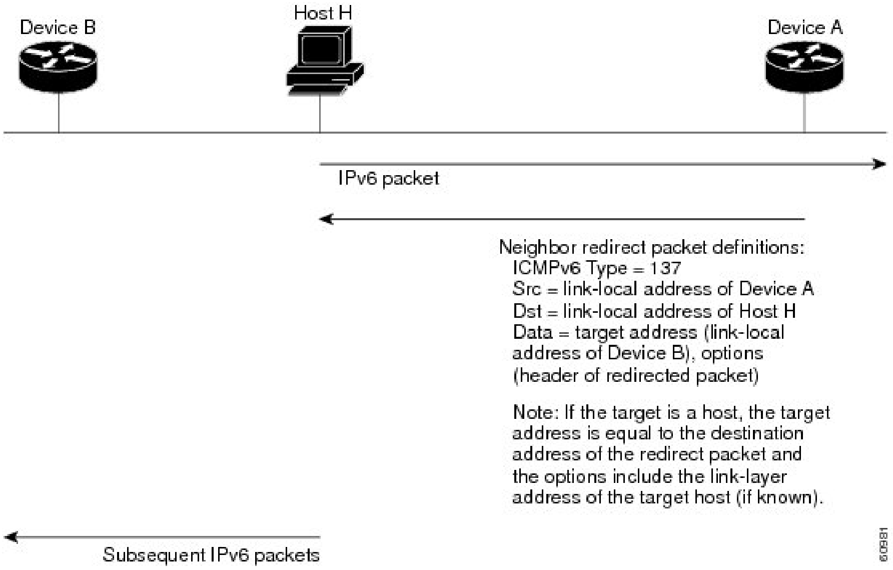
The configured parameters are specific to an interface. The sending of router advertisement messages (with default values) is automatically enabled on Ethernet and FDDI interfaces. For other interface types, the sending of router advertisement messages must be manually configured by using the **no ipv6 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


Device B
Host H
Device A
IPv6 packet
Neighbor redirect packet definitions:
ICMPv6 Type = 137
Src = link-local address of Device A
Dst = link-local address of Host H
Data = target address (link-local address of Device B), options (header of redirected packet)
Note: If the target is a host, the target address is equal to the destination address of the redirect packet and the options include the link-layer address of the target host (if known).
Subsequent IPv6 packets
60981

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 HundredGigE 0/2/0/0: 10.1.1.1/16

interface HundredGigE 0/3/0/0: 10.1.1.2/8

interface HundredGigE 0/4/0/0: 10.2.1.1/16

The IP address on HundredGigE 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 HundredGigE 0/4/0/0 does not conflict with the current highest precedence IP address, the address on HundredGigE 0/4/0/0 is enabled as well.

Recursive Resolution of Conflict Sets

In the example below, the address on the interface in HundredGigE 0/2/0/0 has the highest precedence because it is the lowest rack/slot. However, now the addresses on HundredGigE 0/4/0/0 and HundredGigE 0/5/0/0 also do not conflict with the highest precedence IP addresses on HundredGigE 0/2/0/0. However, the addresses on HundredGigE 0/4/0/0 and HundredGigE 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 HundredGigE 0/4/0/0 is on a lower rack/slot, it is enabled.

interface HundredGigE 0/2/0/0: 10.1.1.1/16

interface HundredGigE 0/3/0/0: 10.1.1.2/8

interface HundredGigE 0/4/0/0: 10.2.1.1/16

interface HundredGigE 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**]
4. route-tag *[route-tag value]*
5. **commit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure**	
Step 2	**interface** *type interface-path-id* **Example:** `RP/0/RP0/CPU0:router(config)# interface hundredgige 0/1/0/1`	Enters interface configuration mode.
Step 3	Do one of the following: • **ipv4 address** *ipv4-address mask* [**secondary**] **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 address for an interface.
Step 4	route-tag *[route-tag value]* **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** *type interface-path-id* **Example:** `RP/0/RP0/CPU0:router(config)# interface hundredgige 0/1/0/1`	Enters interface configuration mode.

Step 3	**ipv4 address** *ipv4-address mask* [**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.
		• 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 **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 FF02:0:0:0:0:1: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** *type interface-path-id* **Example:** `RP/0/RP0/CPU0:router(config)# interface hundredgige 0/1/0/3`	Enters interface configuration mode.

Step 3	Do one of the following:
	• **ipv6 address** *ipv6-prefix* / *prefix-length* [**eui-64**]
	• **ipv6 address** *ipv6-address* **link-local**
	• **ipv6 enable**
	Example:
	`RP/0/RP0/CPU0:router(config-if)# ipv6 address 2001:0DB8:0:1::/64 eui-64`
	`RP/0/RP0/CPU0:router(config-if)# ipv6 address 2001:0DB8:0:1::1/64`
	or
	`RP/0/RP0/CPU0:router(config-if)# ipv6 address FE80::260:3EFF:FE11:6770 link-local`
	or
	`RP/0/RP0/CPU0:router(config-if)# ipv6 enable`
		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.
		• 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.
		or

	Command or Action	Purpose
		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** *type interface-path-id* **Example:** `RP/0/RP0/CPU0:router(config)# interface hundredgige 0/1/0/3`	Enters interface configuration mode.
Step 3	**ipv4 address** *ipv4-address mask* [**secondary**] **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** *type interface-path-id* **Example:** `RP/0/RP0/CPU0:router(config)# interface hundredgige 0/1/0/1`	Specifies the interface type and number, and enters interface configuration mode.
Step 3	**ipv4 address** *ip-address mask* [**secondary**] **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** *ipv6-prefix*/*prefix-length* [**eui-64**] **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. • A slash mark (/) must precede the *prefix-length* , and there is no space between the *ipv6-prefix* and slash mark.
Step 5	**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

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

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure**	

Step 2	Do one of the following:
	• **icmp ipv4 rate-limit unreachable** [**DF**] *milliseconds*
	• **ipv6 icmp error-interval** *milliseconds* [*bucketsize*]
	Example:
	`RP/0/RP0/CPU0:router(config)# icmp ipv4 rate-limit unreachable 1000`
	or
	`RP/0/RP0/CPU0:router(config)# ipv6 icmp error-interval 50 20`
		Limits the rate that IPv4 ICMP destination unreachable messages are generated.
		• 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 *milliseconds* argument specifies the time period between the sending of ICMP destination unreachable messages.
		or
		Configures the interval and bucket size for IPv6 ICMP error messages.
		• The *milliseconds* argument specifies the interval between tokens being added to the bucket.
		• The optional *bucketsize* argument defines the maximum number of tokens stored in the bucket.

Step 3	**commit**	

	Command or Action	Purpose

Step 4	Do one of the following:
	• **show ipv4 traffic** [**brief**]
	• **show ipv6 traffic** [**brief**]
	Example:
	`RP/0/RP0/CPU0:router# show ipv4 traffic`
	or
	`RP/0/RP0/CPU0:router# show ipv6 traffic`
		(Optional) Displays statistics about IPv4 traffic, including ICMP unreachable information.
		• Use the **brief** keyword to display only IPv4 and ICMPv4 traffic statistics.
		or
		(Optional) Displays statistics about IPv6 traffic, including IPv6 ICMP rate-limited counters.
		• Use the **brief** keyword to display only IPv6 and ICMPv6 traffic statistics.

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**	
Step 2	{**ipv4** | **ipv6**} **conflict-policy static** **Example:** `RP/0/RP0/CPU0:router(config)# ipv4 conflict-policy static` or	Sets the conflict policy to static, that is, prevents new interface addresses from affecting the currently running interface.

	Command or Action	Purpose
	`RP/0/RP0/CPU0:router(config)# ipv6 conflict-policy static`	
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:
	`RP/0/RP0/CPU0:router(config)# ipv4 conflict-policy longest-prefix`
	or
	`RP/0/RP0/CPU0:router(config)# ipv6 conflict-policy longest-prefix`
		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	**configure**	

	Command or Action	Purpose

Step 2	**{ ipv4 | ipv6 } conflict-policy highest-ip**
	Example:
	`RP/0/RP0/CPU0:router(config)# ipv4 conflict-policy highest-ip`
	or
	`RP/0/RP0/CPU0:router(config)# ipv6 conflict-policy highest-ip`
		Sets the conflict policy to the highest IP value, that is, the IP address with the highest value gets precedence.

Step 3	**commit**	

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	*Configuring ARP* module in this publication.
Mapping host names to IP addresses	*Host Services and Applications Commands* module in the *IP Addresses and Services Command Reference for Cisco NCS 6000 Series Routers*
Network stack IPv4 and IPv6 commands: complete command syntax, command modes, command history, defaults, usage guidelines, and examples	*Network Stack IPv4 and IPv6 Commands* section in the *IP Addresses and Services Command Reference for Cisco NCS 6000 Series Routers*

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 NCS 6000 Series Routers*.

Feature History for Configuring ARP

Release	Modification
Release 5.0.0	This feature was introduced.

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.

- ARP throttling is not supported.

Note ARP throttling is the rate limiting of ARP packets in Forwarding Information Base (FIB).

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.

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.

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** *ip-address hardware-address encapsulation-type*
   - **arp** *ip-address hardware-address encapsulation-type* **alias**
3. **commit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure**	

Step 2	Do one of the following:
	• **arp** *ip-address hardware-address encapsulation-type*
	• **arp** *ip-address hardware-address encapsulation-type* **alias**
	Example:
	`RP/0/RP0/CPU0:router(config)# arp 192.168.7.19 0800.0900.1834 arpa`
	or
	`RP/0/RP0/CPU0:router(config)# arp 192.168.7.19 0800.0900.1834 arpa alias`
		Creates a static ARP cache entry associating the specified 32-bit IP address with the specified 48-bit hardware address.
		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.

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** *type number* **Example:** `RP/0/RP0/CPU0:router(config)# interface MgmtEth 0//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**	

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 12.1.3.4, 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 recv, 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	*ARP Commands* module in *IP Addresses and Services Command Reference for Cisco NCS 6000 Series Routers*

Related Topic	Document Title
Getting started material	

Related Topic	Document Title
QoS commands: complete command syntax, command modes, command history, defaults, usage guidelines, and examples	*Quality of Service Commands* module in *Modular QoS Command Reference for Cisco NCS 6000 Series Routers*
Class-based traffic shaping, traffic policing, low latency queuing, and MDDR	Configuring Modular Quality of Service Congestion Management module in *Modular QoSConfiguration Guide for Cisco NCS 6000 Series Routers*

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 5.0.0	This feature was introduced .

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

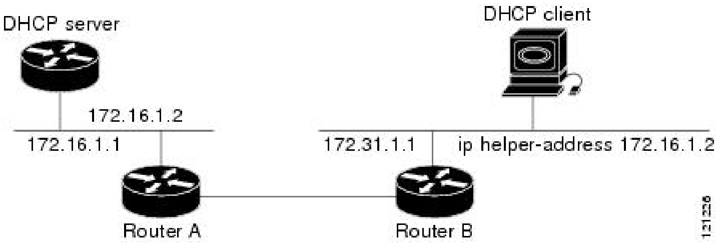
A DHCP relay agent is a host that forwards DHCP packets between clients and servers that do not reside on a shared physical subnet. Relay agent forwarding is distinct from the normal forwarding of an IP router where IP datagrams are switched between networks transparently.

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.

Figure 12: Forwarding UDP Broadcasts to a DHCP Server Using a Helper Address, on page 42 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 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



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.

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 interface-path-id* **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	**interface** *type interface-path-id* **relay profile** *profile-name*
	Example:
	`RP/0/RP0/CPU0:router(config-dhcpv4)# interface`
	`/0 relay profile client`
		Attaches a relay profile to an interface.

Step 4	**commit**	

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 interface-path-id* **none**
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	**interface** *type interface-path-id* **none**
	Example:
	`RP/0/RP0/CPU0:router(config-dhcpv4-relay-profile)# interface`
	`0/1/4/1 none`
		Disables the DHCP relay on the interface.

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**	

	Command or Action	Purpose
Step 2	**dhcp ipv4** **Example:** `RP/0/RP0/CPU0:router(config)# dhcp ipv4`	Enters DHCP IPv4 configuration .
Step 3	**profile** *profile-name* **relay** **Example:** `RP/0/RP0/CPU0:router(config-dhcpv4)# profile client relay`	Enters DHCP IPv4 profile relay .
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. • 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: • Remote ID • Circuit ID **Note** This function is disabled by default.
Step 5	**relay information check** **Example:** `RP/0/RP0/CPU0:router(config-dhcpv4-relay-profile)# relay information check`	(Optional) Configures DHCP to check the relay agent information option in forwarded BOOTREPLY is • By default, DHCP the field in DHCP reply packets, received from the DHCP server. **Note** Use the **relay information check** command to reenable this functionality if the functionality has been disabled.

	Command or Action	Purpose

Step 6	**relay information policy** {**drop** | **keep**}
	Example:
	`RP/0/RP0/CPU0:router(config)# dhcp relay information policy drop`
		(Optional) Configures the reforwarding policy for a DHCP relay agent; that is, whether the relay agent will drop or keep the relay information.

Step 7	**relay information option allow-untrusted** **Example:** `RP/0/RP0/CPU0:router(config-dhcpv4-relay-)# relay information`	(Optional) Configures the DHCP IPv4 Relay not to discard packets that have an existing relay information option and the giaddr set to zero.
Step 8	**commit**	

Configuring Relay Agent Giaddr Policy

This task describes how to configure for that already contain a nonzero giaddr attribute.

SUMMARY STEPS

1. **configure**
2. **dhcp ipv4**
3. **profile 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 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.
		• replace—Replaces the existing giaddr value with a value that it generates.
		• drop—Drops the packet that has an existing nonzero giaddr value.

	Command or Action	Purpose
Step 5	**commit**	

Configuring a DHCP Proxy Profile

The DHCP proxy performs all the functions of a relay and also provides some additional functions. The DHCP proxy conceals DHCP server details from DHCP clients. The DHCP proxy modifies the DHCP replies such that the client considers the proxy to be the server. In this state, the client interacts with the proxy as if it is the DHCP server.

This task describes how to configure and enable the DHCP proxy profile.

SUMMARY STEPS

1. **configure**
2. **dhcp ipv4**
3. **profile** *profile-name* **proxy**
4. **helper-address** *address* [**giaddr** *gateway-address*]
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 .
Step 3	**profile** *profile-name* **proxy** **Example:** `RP/0/RP0/CPU0:router(config-dhcpv4)# profile client proxy`	Enters DHCP IPv4 profile proxy submode.

Step 4	**helper-address** *address* [**giaddr** *gateway-address*]
	Example:
	`RP/0/RP0/CPU0:router(config-dhcpv4-proxy-profile)# helper-address`
	`10.10.1.1`
		Forwards UDP broadcasts, including DHCP.
		• The value of the *address* argument can be a specific DHCP server address or a network address (if other DHCP servers are on the destination network segment). Using the network address enables other servers to respond to DHCP requests.
		• For multiple servers, configure one helper address for each server.

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
```

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  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 HundredGigE 0/1/1/0 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

 !
!
```

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

The following example shows how to configure Cisco IOS XR broadcast flag policy:

```
dhcp ipv4
profile client relay
broadcast-flag policy check
!
```

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	*DHCP Commands* module in the *IP Addresses and Services Command Reference for Cisco NCS 6000 Series Routers*
Information about user groups and task IDs	*Configuring AAA Services* module in the *System Security Configuration Guide for Cisco NCS 6000 Series Routers*

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	*Dynamic Host Configuration 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 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 NCS 6000 Series Routers*

.

Feature History for Implementing Host Services and Applications

Release	Modification
Release 5.0.0	This feature was introduced.

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)—FTP, TFTP 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 *System Management Configuration Guide for the Cisco NCS 6000 Series 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 the Cisco NCS 6000 Series 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.

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) 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. FTP, TFTP can be used to save the core dump to a remote server. See the *System Management Configuration Guide for Cisco NCS 6000 Series Routers* for information on executing a core dump.

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).

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**] [*host-name* | *ip-address*]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**ping** [**ipv4** | **ipv6**] [*host-name* | *ip-address*] **Example:** `RP/0/RP0/CPU0:router# ping`	Starts the ping tool that is used for testing connectivity. **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.

Checking Packet Routes

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

SUMMARY STEPS

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

DETAILED STEPS

	Command or Action	Purpose
Step 1	**traceroute** [**ipv4** | **ipv6**] [*host-name* | *ip-address*] **Example:** `RP/0/RP0/CPU0:router# traceroute`	Traces packet routes through the network. **Note** If you do not enter a hostname or an IP address on the same line as the **traceroute** 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.

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 NCS 6000 Series 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:
	• **domain name** *domain-name*
	• or
	• **domain list** *domain-name*
	Example:
	`RP/0/RP0/CPU0:router(config)# domain name cisco.com`
	`or`
	`RP/0/RP0/CPU0:router(config)# domain list domain1.com`
		Defines a default domain name used to complete unqualified hostnames.

Step 3	**domain name-server** *server-address* **Example:** `RP/0/RP0/CPU0:router(config)# domain name-server 192.168.1.111`	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** {**ipv4** | **ipv6**} **host** *host-name* {*ipv4address* | *ipv6address*} **Example:**	(Optional) Defines a static hostname-to-address mapping in the host cache using . **Note** You can bind up to eight additional associated addresses to a hostname.

	Command or Action	Purpose
	`RP/0/RP0/CPU0:router(config)# domain ipv4 host1 192.168.7.18`	
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.
3. **commit**
4. show cinetd services

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure**	
Step 2	**Example:** `RP/0/RP0/CPU0:router(config)# tftp ipv4 server homedir /tmp access-list listA homedir disk0`	Specifies: • 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	**commit**	
Step 4	show cinetd services **Example:** `RP/0/RP0/CPU0:router# show cinetd services`	Displays the network service for each process. The service column shows TFTP if the TFTP server is configured.

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 implementation 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** *password* **Example:** `RP/0/RP0/CPU0:router(config)# ftp client anonymous-password xxxx`	Specifies the password for anonymous users.
Step 4	**ftp client source-interface** *type* *interface-path-id* **Example:** `RP/0/RP0/CPU0:router(config)# ftp client source-interface HundredGigE 0/1/2/1`	Specifies the source IP address for FTP connections.
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 NCS 6000 Series 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** *type* **Example:** `RP/0/RP0/CPU0:router(config)# tftp client source-interface HundredGigE 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 NCS 6000 Series 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** |] **server max-servers 1**
3. **commit**

DETAILED STEPS

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

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 bost2 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 FTP or TFTP Connections: Example

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

Using FTP

```
configure

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

Using TFTP

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

Additional References

The following sections provide references related to implementing host services and addresses on .

Related Documents

Related Topic	Document Title
Host services and applications commands	*Host Services and Applications Commands* module in *IP Addresses and Services Command Reference for Cisco NCS 6000 Series Routers*

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

Implementing Access Lists and Prefix Lists

An access control list (ACL) consists of one or more access control entries (ACE) that collectively define the network traffic profile. This profile can then be referenced by Cisco IOS XR softwarefeatures such as traffic filtering, route filtering, QoS classification, and access control. Each ACL includes an action element (permit or deny) and a filter element based on criteria such as source address, destination address, protocol, and protocol-specific parameters.

Prefix lists are used in route maps and route filtering operations and can be used as an alternative to access lists in many Border Gateway Protocol (BGP) route filtering commands. A prefix is a portion of an IP address, starting from the far left bit of the far left octet. By specifying exactly how many bits of an address belong to a prefix, you can then use prefixes to aggregate addresses and perform some function on them, such as redistribution (filter routing updates).

Note For a complete description of the access list and prefix list commands listed in this module, refer to the *IP Addresses and Services Command Reference for Cisco NCS 6000 Series Routers*.

Feature History for Implementing Access Lists and Prefix Lists

Release	Modification
Release 5.0.0	This feature was introduced.

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

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

- Layer 2/Layer 3 ACLs are not supported on Layer 2 interfaces.
- 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 tcam summary all acl all location** and **show pfilter-ea fea summary location** commands to view the TCAM utilization.

- Video Monitoring is not supported through ACLs on IPv6 interfaces.

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.

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.

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.
- 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, 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 is not 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. 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.

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 is not 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 is when a threshold is not 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 should not be used as a billing tool or as an accurate source of the number of matches to an access list.

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	For an access-list entry containing only Layer 3 information: • The entry is applied to non-fragmented packets, initial fragments, and non-initial fragments. For an access-list entry containing Layer 3 and Layer 4 information: • The entry is applied to non-fragmented packets and initial fragments. • 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 • 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. **Note** Note that the deny statements are handled differently for non-initial fragments versus non-fragmented or initial fragments.
...the **fragments** keyword and all of the access-list entry information matches	The access-list entry is applied only to non-initial fragments. **Note** The **fragments** keyword cannot be configured for an access-list entry that contains any Layer 4 information.

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 80” 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 fowarding 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**} **ipv4** *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* [*sequence-number*] [**hardware** {**ingress** | **egress**} [**interface** *interface-path-id* | **sequence** *number* | **location** *node-id*] | [**usage** {**pfilter location** *node-id*}] [**summary** [*access-list-name*]] [**maximum** [**detail**]]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure**	

Step 2	{**ipv4** | **ipv6**} **access-list** *name*
	Example:
	`RP/0/RP0/CPU0:router(config)# ipv4 access-list acl_1`
	or
	`RP/0/RP0/CPU0:router(config)# ipv6 access-list acl_2`
		Enters either IPv4 or IPv6 access list configuration mode and configures the named access list.

Step 3	[*sequence-number*] **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 a **permit** or **deny** statement in a named access list. • 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	Do one of the following: • [*sequence-number*]{**permit** | **deny**} **ipv4** *source source-wildcard destination destination-wildcard* [**precedence** *precedence*] [**dscp** *dscp*] [**fragments**] [**log** | **log-input**]	Specifies one or more conditions allowed or denied in IPv4 access list acl_1. • The optional **log** keyword causes an information logging message about the packet that matches the entry to be sent to the console.

	Command or Action	Purpose

	• [*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**]
	Example:
	`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`
	or
	`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`
		• The optional **log-input** keyword provides the same function as the **log** keyword, except that the logging message also includes the input interface.
		or
		Specifies one or more conditions allowed or denied in IPv6 access list acl_2.
		• 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.
		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 an access list.
Step 6	**commit**	
Step 7	**show access-lists** {**ipv4** | **ipv6**} [*access-list-name* [*sequence-number*] [**hardware** {**ingress** | **egress**} [**interface** *interface-path-id* | **sequence** *number* | **location** *node-id*] | [**usage** {**pfilter location** *node-id*}] [**summary** [*access-list-name*]] [**maximum** [**detail**]] **Example:** `RP/0/RP0/CPU0:router# show access-lists ipv4 acl_1`	(Optional) Displays the contents of current IPv4 or IPv6 access lists. • Use the *access-list-name* 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 80 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 XR Config 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**	
Step 2	**interface** *type interface-path-id* **Example:** `RP/0/RP0/CPU0:router(config)# interface HundredGigE 0/2/0/2`	Configures an interface and enters interface configuration mode. • The *type* argument specifies an interface type. For more information on interface types, use the question mark (?) online help function.

	Command or Action	Purpose
		• 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	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**]
	Example:
	`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`
		Controls access to an interface.
		• Use the *access-list-name* 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.
		  • Hardware counters are automatically enabled for IPv6 access groups.
		• Use the **interface-statistics** keyword to specify per-interface statistics in the hardware.
		This example applies filters on packets inbound and outbound from HundredGigE interface 0/2/0/2.

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

1. **configure**
2. **line** {**console** | **default** | **template** *template-name*}
3. **access-class** *list-name* {**in** | **out**}
4. **commit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure**	
Step 2	**line** {**console** | **default** | **template** *template-name*} **Example:**	Specifies either the console, default, or a user-defined line template and enters line template configuration mode.

	Command or Action	Purpose
	`RP/0/RP0/CPU0:router(config)# line default`	• Line templates are a collection of attributes used to configure and manage physical terminal line connections (the console port) and vty connections. The following templates are available in Cisco IOS XR software: • 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** *list-name* {**in** | **out**}
	Example:
	`RP/0/RP0/CPU0:router(config-line)# access-class acl_2 out`
		Restricts incoming and outgoing connections using an IPv4 or IPv6 access list.
		• In the example, outgoing connections for the default line template are filtered using the IPv6 access list acl_2.

Step 4	**commit**	

Configuring Prefix Lists

This task configures an IPv4 or IPv6 prefix list.

SUMMARY STEPS

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

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure**	

	Command or Action	Purpose

Step 2	{**ipv4** | **ipv6**} **prefix-list** *name*
	Example:
	`RP/0/RP0/CPU0:router(config)# ipv4 prefix-list pfx_1`
	or
	`RP/0/RP0/CPU0:router(config)# ipv6 prefix-list pfx_2`
		Enters either IPv4 or IPv6 prefix list configuration mode and configures the named prefix list.
		• To create a prefix list, you must enter at least one **permit** or **deny** clause.
		• Use the **no** {**ipv4** | **ipv6**} **prefix-list** *name* command to remove all entries in a prefix list.

Step 3	[*sequence-number*] **remark** *remark*
	Example:
	`RP/0/RP0/CPU0:router(config-ipv4_pfx)# 10 remark Deny all routes with a prefix of 10/8`
	`RP/0/RP0/CPU0:router(config-ipv4_pfx)# 20 deny 10.0.0.0/8 le 32`
		(Optional) Allows you to comment about the following **permit** or **deny** statement in a named prefix list.
		• 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	[*sequence-number*] {**permit** | **deny**} *network/length* [**ge** *value*] [**le** *value*] [**eq** *value*] **Example:** `RP/0/RP0/CPU0:router(config-ipv6_pfx)# 20 deny 128.0.0.0/8 eq 24`	Specifies one or more conditions allowed or denied in the named prefix list. • 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** *sequence-number* command to delete an entry.	Allows you to revise a prefix list.
Step 6	**commit**	

Step 7	Do one of the following:
	• **show prefix-list ipv4** [*name*] [*sequence-number*]
	• **show prefix-list ipv6** [*name*] [*sequence-number*] [*summary*]
	Example:
	`RP/0/RP0/CPU0:router# show prefix-list ipv4 pfx_1`
	or
	`RP/0/RP0/CPU0:router# show prefix-list ipv6 pfx_2 summary`
		(Optional) Displays the contents of current IPv4 or IPv6 prefix lists.
		• Use the *name* argument to display the contents of a specific prefix list.
		• Use the *sequence-number* argument to specify the sequence number of the prefix-list entry.
		• Use the **summary** keyword to display summary output of prefix-list contents.

Step 8	**clear prefix-list** {**ipv4** | **ipv6**} *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 *hit count* 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* [*sequence-number*] [**hardware** {**ingress** | **egress**} [**interface** *interface-path-id* | **sequence** *number* | **location** *node-id*] | [**usage** {**pfilter location** *node-id*}] [**summary** [*access-list-name*]] [**maximum** [**detail**]]

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	[*sequence-number*] **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. • 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	[*sequence-number*] {**permit** | **deny**} *source* [*source-wildcard*] [**log** | **log-input**]
	Example:
	`RP/0/RP0/CPU0:router(config-ipv4-acl)# 20 permit 172.16.0.0 0.0.255.255`
	or
	`RRP/0/RP0/CPU0:routerrouter(config-ipv4-acl)# 30 deny 192.168.34.0 0.0.0.255`
		Specifies one or more conditions allowed or denied, which determines whether the packet is passed or dropped.
		• Use the *source* argument to specify the number of network or host from which the packet is being sent.
		• Use the optional *source-wildcard* 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.

	Command or Action	Purpose
		• 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	**show access-lists** {**ipv4** | **ipv6**} [*access-list-name* [*sequence-number*] [**hardware** {**ingress** | **egress**} [**interface** *interface-path-id* | **sequence** *number* | **location** *node-id*] | [**usage** {**pfilter location** *node-id*}] [**summary** [*access-list-name*]] [**maximum** [**detail**]] **Example:** `RP/0/RP0/CPU0:router# show access-lists ipv4 acl_1`	(Optional) Displays the contents of the named IPv4 access list. • 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 80” 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* [*sequence-number*] [**hardware** {**ingress** | **egress**} [**interface** *interface-path-id* | **sequence** *number* | **location** *node-id*] | [**usage** {**pfilter location** *node-id*}] [**summary** [*access-list-name*]] [**maximum** [**detail**]]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**copy access-list** {**ipv4** | **ipv6**}*source-acl destination-acl* **Example:** `RP/0/RP0/CPU0:router# copy ipv6 access-list list-1 list-2`	Creates a copy of an existing IPv4 or IPv6 access list. • Use the *source-acl* argument to specify the name of the access list to be copied. • Use the *destination-acl* argument to specify where to copy the contents of the source access list. • The *destination-acl* argument must be a unique name; if the *destination-acl* argument name

	Command or Action	Purpose
		exists for an access list, the access list is not copied.
Step 2	**show access-lists** {**ipv4** | **ipv6**} [*access-list-name* [*sequence-number*] [**hardware** {**ingress** | **egress**} [**interface** *interface-path-id* | **sequence** *number* | **location** *node-id*] | [**usage** {**pfilter location** *node-id*}] [**summary** [*access-list-name*]] [**maximum** [**detail**]] **Example:** `RP/0/RP0/CPU0:router# show access-lists ipv4 list-2`	(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.

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**} [**ipv4**] *source source-wildcard destination destination-wildcard* [**precedence** *precedence*] [**dscp** *dscp*] [**fragments**] [**log** | **log-input**]
   - [*sequence-number*] {**permit** | **deny**} *protocol* {*source source-wildcard* | **any** | **host** *source*} [*operator* {*port* | *protocol-port*}] {*destinationdestination-wildcard* | **any** | **host** *destination*} [*operator* {*port* | *protocol-port*}] [**dscp** *value*] [**routing**] [**authen**] [**destopts**] [**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* [*sequence-number*] [**hardware** {**ingress** | **egress**} [**interface** *interface-path-id* | **sequence** *number* | **location** *node-id*] | [**usage** {**pfilter location** *node-id*}] [**summary** [*access-list-name*]] [**maximum** [**detail**]]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**resequence access-list** {**ipv4** | **ipv6**} *name* [*base* [*increment*]] **Example:** `RP/0/RP0/CPU0:router# resequence access-list ipv4 acl_3 20 15`	(Optional) Resequences the specified IPv4 or IPv6 access list using the starting sequence number and the increment of sequence numbers. • This example resequences an IPv4 access list named acl_3. 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	{**ipv4** | **ipv6**} **access-list** *name*
	Example:
	`RP/0/RP0/CPU0:router(config)# ipv4 access-list acl_1`
	or
	`RP/0/RP0/CPU0:router(config)# ipv6 access-list acl_2`
		Enters either IPv4 or IPv6 access list configuration mode and configures the named access list.

Step 4	Do one of the following: • [*sequence-number*] {**permit** | **deny**} [**ipv4**] *source source-wildcard destination destination-wildcard* [**precedence** *precedence*] [**dscp** *dscp*] [**fragments**] [**log** | **log-input**] • [*sequence-number*] {**permit** | **deny**} *protocol* {*source source-wildcard* | **any** | **host** *source*} [*operator* {*port* | *protocol-port*}] {*destinationdestination-wildcard* | **any** | **host** *destination*} [*operator* {*port* | *protocol-port*}] [**dscp** *value*] [**routing**] [**authen**] [**destopts**] [**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**] **Example:** `RP/0/RP0/CPU0:router(config-ipv4-acl)# 10 permit 172.16.0.0 0.0.255.255`	Specifies one or more conditions allowed or denied in IPv4 access list acl_1. • 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. or Specifies one or more conditions allowed or denied in IPv6 access list acl_2. • 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-ipv4-acl)# 20 deny 192.168.34.0 0.0.0.255`
	or
	`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`
		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* [*sequence-number*] [**hardware** {**ingress** | **egress**} [**interface** *interface-path-id* | **sequence** *number* | **location** *node-id*] | [**usage** {**pfilter location** *node-id*}] [**summary** [*access-list-name*]] [**maximum** [**detail**]] **Example:** `RP/0/RP0/CPU0:router# show access-lists ipv4 acl_1`	(Optional) Displays the contents of a named IPv4 or IPv6 access list. • 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 80" 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:**	Creates a copy of an existing IPv4 or IPv6 prefix list. • Use the *source-name* argument to specify the name of the prefix list to be copied and the *destination-name*

	Command or Action	Purpose
	`RP/0/RP0/CPU0:router# copy prefix-list ipv6 list_1 list_2`	argument to specify where to copy the contents of the source prefix list. • The *destination-name* argument must be a unique name; if the *destination-name* argument name exists for a prefix list, the prefix list is not copied.
Step 2	Do one of the following: • **show prefix-list ipv4** [*name* [*sequence-number*]] [**summary**] • **show prefix-list ipv6** [*name* [*sequence-number*]] [**summary**] **Example:** `RP/0/RP0/CPU0:router# show prefix-list ipv6 list_2`	(Optional) Displays the contents of current IPv4 or IPv6 prefix lists. • 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

1. **resequence prefix-list ipv4** *name* [*base* [*increment*]]
2. **configure**
3. {**ipv4** | **ipv6**} **prefix-list** *name*
4. [*sequence-number*] {**permit** | **deny**} *network/length* [**ge** *value*] [**le** *value*] [**eq** *value*]
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. 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** *name* [*base* [*increment*]] **Example:** `RP/0/RP0/CPU0:router# resequence prefix-list ipv4 pfx_1 10 15`	(Optional) Resequences the named IPv4 prefix list using the starting sequence number and the increment of sequence numbers. • 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** *name* **Example:** `RP/0/RP0/CPU0:router(config)# ipv6 prefix-list pfx_2`	Enters either IPv4 or IPv6 prefix list configuration mode and configures the named prefix list.
Step 4	[*sequence-number*] {**permit** | **deny**} *network/length* [**ge** *value*] [**le** *value*] [**eq** *value*] **Example:** `RP/0/RP0/CPU0:router(config-ipv6_pfx)# 15 deny 128.0.0.0/8 eq 24`	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: • **show prefix-list ipv4** [*name*] [*sequence-number*] • **show prefix-list ipv6** [*name*] [*sequence-number*] [**summary**] **Example:** `RP/0/RP0/CPU0:router# show prefix-list ipv6 pfx_2`	(Optional) Displays the contents of current IPv4 or IPv6 prefix lists. • 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** [**ipv4** *ipv4-address1*] **nexthop2**[**ipv4** *ipv4-address2*] **nexthop3**[**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** *name* **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	[*sequence-number*] **permit** *protocol source source-wildcard destination destination-wildcard* [**precedence** *precedence*] [**[default] nexthop1** [**ipv4** *ipv4-address1*] **nexthop2**[**ipv4** *ipv4-address2*] **nexthop3**[**ipv4** *ipv4-address3*]] [**dscp** *dscp*] [**fragments**] [**log** | **log-input**] [[**ttl** *ttl* [*value1 ... value2*]]
	Example:
	`RP/0/RP0/CPU0:router(config-ipv4-acl)# 10 permit ipv4 10.0.0.0 0.255.255.255 any nexthop 50.1.1.2`
	`RP/0/RP0/CPU0:router(config-ipv4-acl)# 15 permit ipv4 30.2.1.0 0.0.0.255 any`
	`RP/0/RP0/CPU0:router(config-ipv4-acl)# 20 permit ipv4 30.2.0.0 0.0.255.255 any nexthop 40.1.1.2`
	`RP/0/RP0/CPU0:router(config-ipv4-acl)# 25 permit ipv4 any any`
		Sets the conditions for an IPv4 access list. The configuration example shows how to configure ACL-based forwarding with security ACL.
		• 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** [[*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 security-abf-acl`	Displays the information for ACL software.

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:** `RP/0/RP0/CPU0:router(config)# ipv6 access-list security-abf-acl`	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:** `RP/0/RP0/CPU0:router(config-ipv6-acl)# 10 permit ipv6 host 100:1:1:2:3::1 host 10:11:12::2 nexthop1 ipv6 195:1:1:200:5ff:fe00:0`	Sets the conditions for an IPv6 access list. The configuration example shows how to configure pure ACL-based forwarding for ACL. • The **nexthop** keyword forwards the specified next hop for this entry.
Step 4	**commit**	

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	**configure**	
Step 2	{**ipv4** } **access-list** *name* **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	[*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*]]
	Example:
	`RP/0/RP0/CPU0:router(config-ipv4-acl)# 10 permit ipv4 10.0.0.0 0.255.255.255 any nexthop 50.1.1.2`
	`RP/0/RP0/CPU0:router(config-ipv4-acl)# 15 permit ipv4 30.2.1.0 0.0.0.255 any`
	`RP/0/RP0/CPU0:router(config-ipv4-acl)# 20 permit ipv4 30.2.0.0 0.0.255.255 any nexthop 40.1.1.2`
	`RP/0/RP0/CPU0:router(config-ipv4-acl)# 25 permit ipv4 any any`
		Sets the conditions for an IPv4 or an IPv6 access list. The configuration example shows how to configure pure ACL-based forwarding for ACL.
		• The **nexthop** keyword forwards the specified next hop for this entry.

Step 4	**commit**	
Step 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*}]] **Example:** `RP/0/RP0/CPU0:router# show access-lists ipv4 security-abf-acl`	Displays the information for ACL software.

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 , and increment value is . 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
 ipv4 access-list acl_1
 end
resequence ipv4 access-list acl_1 10 20

show access-lists ipv4 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 permit ip host 172.16.2.2 host 10.3.3.12
110 permit ip host 10.3.3.3 any log
130 permit tcp host 10.3.3.3 host 10.1.2.2
150 permit ip any any
```

Adding Entries with Sequence Numbers: Example

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

```
show access-lists ipv4 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
```

```
show access-lists ipv4 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
 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

show access-lists ipv4 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 0.0.0.255
end

show access-lists ipv4 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
```

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	*Access List Commands* module in *IP Addresses and Services Command Reference for Cisco NCS 6000 Series Routers*
Prefix list commands: complete command syntax, command modes, command history, defaults, usage guidelines, and examples	*Prefix List Commands* module in *IP Addresses and Services Command Reference for Cisco NCS 6000 Series Routers*

Related Topic	Document Title
Terminal services commands: complete command syntax, command modes, command history, defaults, usage guidelines, and examples	*Terminal Services Commands* module in *System Management Command Reference for Cisco NCS 6000 Series Routers*

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 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, refer to the Related Documents, on page 109 section of this module.

Feature History for Implementing CEF

Release	Modification
Release 5.0.0	This feature was introduced.
Release 5.0.1	Support for Loose and Strict uRPF was added.

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:

- Multipath support
- 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 softwareCEF 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 ( ) and each MSC.

The FIB on each node processes Routing Information Base (RIB) updates, performing route resolution and maintaining FIB tables independently in the 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.

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, 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

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
OOR state: GREEN. Number of OOR attributes: 0

BGP Attribute ID: 0x6, Local Attribute ID: 0x1
    Aspath      :    2
    Community   :
    Origin AS   :    2
    Next Hop AS :    2
```

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:**	Displays the IPv4 or IPv6 CEF table. The next hop and forwarding interface are displayed for each prefix.

	Command or Action	Purpose
	`RP/0/RP0/CPU0:router# show cef ipv4`	**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	show adjacency detail **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 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** *seconds* **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 verify unicast source reachable-via** {**any** | **rx**} [**allow-default**] [**allow-self-ping**]
4. **commit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure**	
Step 2	**interface** *type interface-path-id* **Example:** `RP/0/RP0/CPU0:router(config)# interface TenGigE 0/1/0/0`	Enters interface configuration mode.
Step 3	**ipv4 verify unicast source reachable-via** {**any** | **rx**} [**allow-default**] [**allow-self-ping**] **Example:** `RP/0/RP0/CPU0:router(config-if)# ipv4 verify unicast source reachable-via rx`	Enables IPv4 uRPF checking. • 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.
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:** `RP/0/RP0/CPU0:router(config)# rp mgmtethernet forwarding`	Enables switching from the MSC to the route processor Management Ethernet interfaces.
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:
	`RP/0/RP0/CPU0:router(config)# cef load-balancing fields L4`
		Configures the hashing algorithm that is used for load balancing during forwarding. The example shows that the L4 field is selected.
		• Use the **L3** keyword to specify the Layer 3 load-balancing for the hash
		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.
		• Use the **L4** keyword to specify the Layer 3 and Layer 4 load-balancing for the hash algorithm.
		For a list of the inputs for Layer 3 and Layer 4, see Per-Flow Load Balancing, on page 100.

Step 3	**commit**	

	Command or Action	Purpose
Step 4	**show cef** {**ipv4** | **ipv6**} **summary** [**location** *node-id*] **Example:** `RP/0/RP0/CPU0:router# show cef ipv4 summary`	Displays the load balancing field for the IPv4 or IPv6 CEF table. • (Optional) Use the **location** keyword display a summary of the IPv4 CEF table for the designated node. The *node-id* argument is entered in the *rack/slot/module* 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

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 XR 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 XR 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 XR 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 HundredGigE 0/1/0/4
```

```
22.6.1.9/32 version 0, internal 0x40040001 (0x78439fd0) [3], 0x0 (0x78aaf928), 0x4400 (0x78ed62d0)
 remote adjacency to HundredGigE0/1/4/4  Prefix Len 32, traffic index 0, precedence routine (0)
   via HundredGigE0/1/4/4
```

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 XR 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 HundredGigE 0/1/0/4


22:6:1::/64, version 0, internal 0x40000001 (0x7846c048) [3], 0x0 (0x78aea3d0), 0x0 (0x0)  remote
adjacency to HundredGigE0/1/4/4  Prefix Len 64, traffic index 0, precedence routine (0)
   via HundredGigE0/1/4/4
```

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** {*attribute-id* | *local-attribute-id* }
	Example:
	`RP/0/RP0/CPU0:router(config)# cef bgp attribute`
		Configures a CEF BGP attribute.

Step 3	**commit**	

Configuration Examples for Implementing CEF on Routers Software

This section provides the following configuration examples:

Configuring Unicast RPF Checking: Example

The following example shows how to configure unicast RPF checking:

```
configure
interface TenGigE 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_RP0_CPU0

  Load balancing: L3
  Tableid 0xe0000000 (0x9cbb51b0),  Flags 0x2031
   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
  0 route delete cache elements
  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

```
Router ID is 1.10.10.10

IP CEF with switching (Table Version 0) for node0_RP0_CPU0

  Load balancing: L4
  Tableid 0xe0000000 (0x89bba258), Flags 0x2031
  Refcount 16
  5 routes, 0 protected, 0 reresolve, 0 unresolved (0 old, 0 new), 680 bytes
  5 load sharing elements, 1860 bytes, 0 references
  0 shared load sharing elements, 0 bytes
  5 exclusive load sharing elements, 1860 bytes
  0 route delete cache elements
  0 local route bufs received, 1 remote route bufs received,  0 mix bufs received
  0 local routes, 0 remote routes
  0 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, 0 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 next hops
  0 incomplete next hops

0 PD backwalks on LDIs with backup path
```

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}
```

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	*Cisco Express Forwarding Commands* module in *IP Addresses and Services Command Reference for Cisco NCS 6000 Series Routers*
BGP commands: complete command syntax, command modes, command history, defaults, usage guidelines, and examples	*BGP Commands* module in the *Routing Command Reference for Cisco NCS 6000 Series Routers*
Link Bundling Commands: complete command syntax, command modes, command history, defaults, usage guidelines, and examples	*Link Bundling Commands* module in the *Interface and Hardware Component Command Reference for the Cisco NCS 6000 Series Routers*

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 8

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 5.0.0	This feature was introduced.

Note GLBP is not supported on ASR9k.

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 (BVIs).

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.

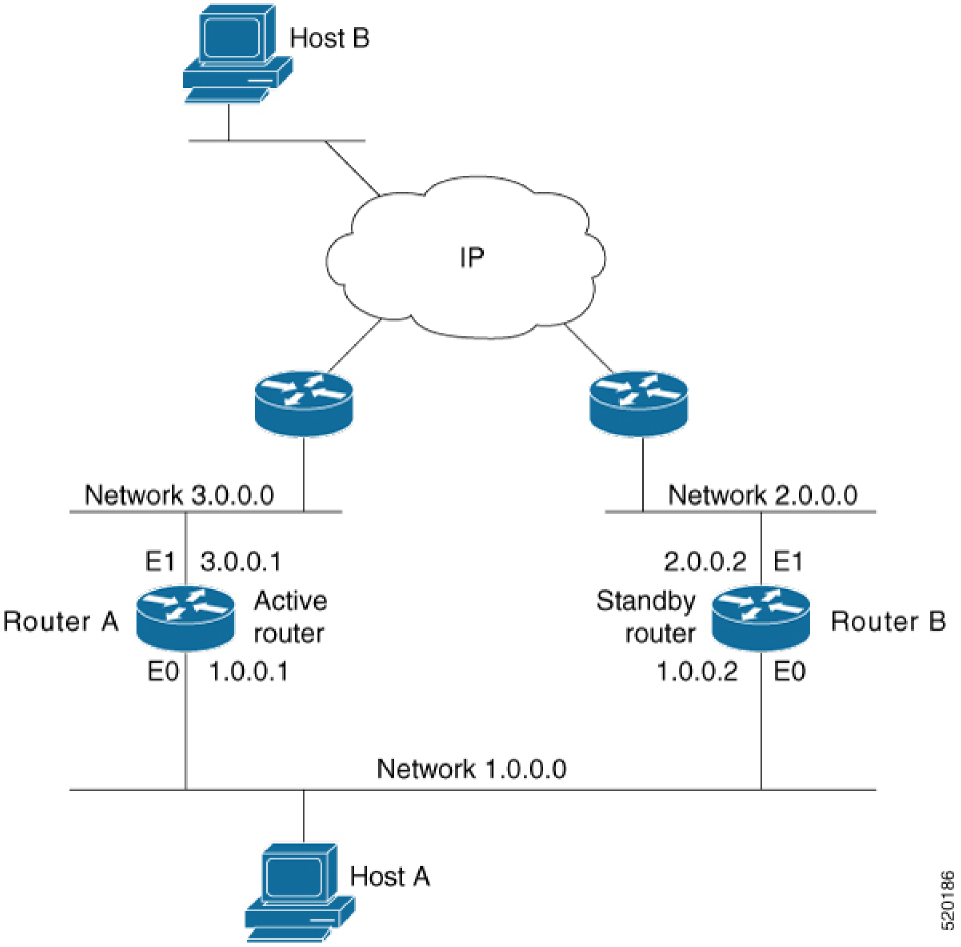
HSRP Groups

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 13: Routers Configured as an HSRP Group, on page 113 shows routers configured as members of a single HSRP group.

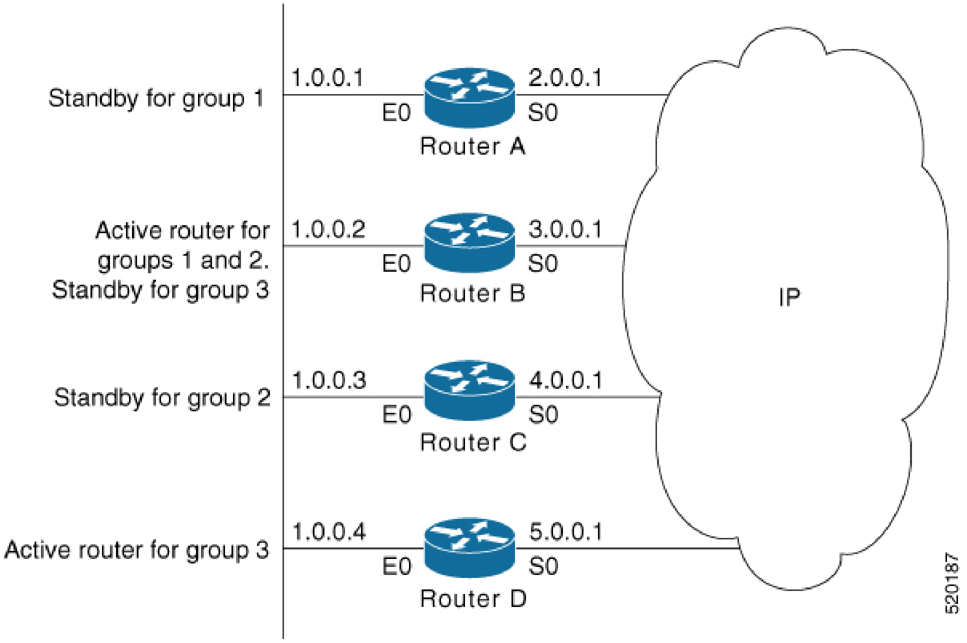
Figure 13: Routers Configured as an HSRP Group



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 14: Routers Configured as Members of Multiple HSRP Groups, on page 114shows routers configured as members of multiple HSRP groups.

Figure 14: Routers Configured as Members of Multiple HSRP Groups



In Figure 14: Routers Configured as Members of Multiple HSRP Groups, on page 114, 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. **hsrp** [*group-number*] **ipv4** [ip-address [secondary]]
5. **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 HundredGigE 0/2/0/1`	Enables HSRP interface configuration mode on a specific interface.
Step 4	**hsrp** [*group-number*] **ipv4** [ip-address [secondary]] **Example:** `RP/0/RP0/CPU0:router(config-hsrp)# hsrp 1 ipv4`	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.
Step 5	**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. **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.

	Command or Action	Purpose
Step 3	**interface** *type interface-path-id* **Example:** `RP/0/RP0/CPU0:router(config-hsrp)# interface HundredGigE 0/2/0/1`	Enables HSRP interface configuration mode on a specific interface.
Step 4	**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. **hsrp** [*group-number*] **priority** *priority*
6. **hsrp** [*group-number*] **track** *type* [*priority-decrement*]
7. **hsrp** [*group-number*] **preempt** [**delay** *seconds*]
8. **hsrp** [*group-number*] **authentication** *string*
9. **hsrp** [*group-number*] **mac-address** *address*
10. **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 HundredGigE  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.

	Command or Action	Purpose
		• 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	**hsrp** [*group-number*] **priority** *priority* **Example:** `RP/0/RP0/CPU0:router(config-hsrp-if)# hsrp priority 100`	(Optional) Configures HSRP priority. • If you do not specify the *group-number*, the configuration applies to all HSRP groups on the router. • 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 **hsrp track** command and another interface on the device goes down. • If preemption is not enabled using the **hsrp** [*group-number*] **preempt** command, the router may not become active even though it might have a higher priority than other HSRP routers. • To restore the default HSRP priority values, use the **no hsrp** command.
Step 6	**hsrp** [*group-number*] **track** *type* [*priority-decrement*] **Example:** `RP/0/RP0/CPU0:router(config-hsrp-if)# hsrp track TenGigE 0/3/0/1`	(Optional) Configures an interface so that the Hot Standby priority changes on the basis of the availability of other interfaces. • If you do not specify the *group-number*, the configuration applies to all HSRP groups on the router. • 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 *priority-decrement* argument specifies by how much to decrement the Hot Standby priority when a tracked interface goes down. When the tracked

	Command or Action	Purpose
		interface comes back up, the priority is incrementally increased by the same amount. • When multiple tracked interfaces are down and the *priority-decrement* 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 **hsrp 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 **hsrp 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 hsrp** command.
Step 7	**hsrp** [*group-number*] **preempt** [**delay** *seconds*] **Example:** `RP/0/RP0/CPU0:router(config-hsrp-if)# hsrp preempt`	(Optional) Configures HSRP preemption and preemption delay. • If you do not specify a value for *group-number*, the configuration applies to all HSRP groups on the router. • When you configure preemption and preemption delay with the **hsrp 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 **hsrp 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). • 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 *delay seconds* 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 **hsrp timers** command), regardless of the preempt delay seconds value. • To restore the default HSRP preemption and preemption delay values, use the **no hsrp** command.

	Command or Action	Purpose
Step 8	**hsrp** [*group-number*] **authentication** *string* **Example:** `RP/0/RP0/CPU0:router(config-hsrp-if)# hsrp 1 authentication company1`	(Optional) Configures an authentication string for the Hot Standby Router Protocol (HSRP). • If you do not specify a value for *group-number*, the configuration applies to all HSRP groups on the router. • 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 hsrp** command.
Step 9	**hsrp** [*group-number*] **mac-address** *address* **Example:** `RP/0/RP0/CPU0:router(config-hsrp-if)# hsrp 5 mac-address 4000.1000.1060`	(Optional) Specifies a virtual MAC address for the HSRP. • If you do not specify a value for the *group-number* argument, the configuration applies to all HSRP groups on the router. • 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 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 **hsrp mac-address** command to specify the virtual MAC address. • The MAC address specified is used as the virtual MAC address when the router is active.

	Command or Action	Purpose
		• The **hsrp 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 **hsrp mac-address** command in the routers to set the virtual MAC address to the value used in the end nodes. • Enter the **no hsrp** [*group-number*] **mac-address** command to revert to the standard virtual MAC address (0000.0C07.ACn).
Step 10	**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. **hsrp** [*group-number*] **ipv4** [ip-address [secondary]]
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** *type interface-path-id* **Example:** `RP/0/RP0/CPU0:router(config-hsrp)# interface HundredGigE 0/2/0/1`	Enables HSRP interface configuration mode on a specific interface.
Step 4	**hsrp delay minimum** *seconds* **reload** *seconds* **Example:**	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

	Command or Action	Purpose
	`RP/0/RP0/CPU0:router(config-hsrp-if)#hsrp delay minimum 2 reload 10`	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	**hsrp** [*group-number*] **ipv4** [ip-address [secondary]] **Example:** `RP/0/RP0/CPU0:router(config-hsrp)# hsrp 1 ipv4`	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.
Step 6	**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. **hsrp** [*group-number*] **ipv4** [ip-address [secondary]]
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** *type interface-path-id* **Example:** `RP/0/RP0/CPU0:router(config-hsrp)# interface HundredGigE 0/2/0/1`	Enables HSRP interface configuration mode on a specific interface.

	Command or Action	Purpose
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. • 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 reenable 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	**hsrp** [*group-number*] **ipv4** [ip-address [secondary]] **Example:** `RP/0/RP0/CPU0:router(config-hsrp)# hsrp 1 ipv4`	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.
Step 6	**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. **hsrp** [*group number*] **bfd fast-detect**
5. **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 HundredGigE 0/2/0/1`	Enables HSRP interface configuration mode on a specific interface.
Step 4	**hsrp** [*group number*] **bfd fast-detect** **Example:** `RP/0/RP0/CPU0:router(config-hsrp-if)# hsrp 1 bfd fast-detect`	Enables fast detection on a specific interface.
Step 5	**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. **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 HundredGigE 0/2/0/1`	Enables HSRP interface configuration mode on a specific interface.
Step 4	**hsrp bfd minimum-interval** *interval* **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	**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. **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 HundredGigE 0/2/0/1`	Enables HSRP interface configuration mode on a specific interface.
Step 4	**hsrp bfd multiplier** *multiplier* **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	**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 HundredGigE0/2/0/1
hsrp 1 ipv4 1.0.0.5
commit
hsrp 1 timers 100 200
hsrp 1 preempt delay 500
hsrp priority 20
hsrp track HundredGigE 0/2/0/2
hsrp 1 authentication company0
hsrp use-bia
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 HundredGigE 0/2/0/3
hsrp 1 ipv4 1.0.0.5
hsrp 1 priority 20
hsrp 1 preempt
hsrp 1 authentication sclara
hsrp 2 ipv4 1.0.0.6
hsrp 2 priority 110
hsrp 2 preempt
hsrp 2 authentication mtview
hsrp 3 ipv4 1.0.0.7
hsrp 3 preempt
hsrp 3 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	*Quality of Service Commands on Modular QoS Command Reference for Cisco NCS 6000 Series Routers*
Class-based traffic shaping, traffic policing, low-latency queuing, and Modified Deficit Round Robin (MDRR)	*Configuring Modular Quality of Service Congestion Management on Modular QoSConfiguration Guide for Cisco NCS 6000 Series Routers*
WRED, RED, and tail drop	*Configuring Modular QoS Congestion Avoidance on Modular QoSConfiguration Guide for Cisco NCS 6000 Series Routers*
HSRP commands	*HSRP Commands on IP Addresses and Services Command Reference for Cisco NCS 6000 Series Routers*
Information about user groups and task IDs	*Configuring AAA Services on System Security Configuration Guide for Cisco NCS 6000 Series Routers*

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

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

CHAPTER 9

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 NCS 6000 Series Routers*.

Feature History for Implementing LPTS

Release	Modification
Release 5.0.0	LPTS was introduced.
Release 5.2.1	Excessive ARP Punt Protection was supported.
Release 5.2.5	Excessive Punt Flow Trap Interface-based Flow feature was introduced.

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). Pre-IFIB (PIFIB), which is an abbreviated copy of IFIB, is maintained by port arbitrator on route processor. The line card also downloads the PIFIB for fast lookup. While IFIB is only present on RP, PIFIB is present on both RP and LCs. The entries in PIFIB are used for a single lookup with an exact match. The IFIB, along with PIFIB are 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.

Excessive ARP Punt Protection

The Excessive ARP Punt Protection feature attempts to identify and mitigate control packet traffic from remote devices that send more than their allocated share of ARP control packet traffic. A remote device can be a device on a VLAN interface.

When remote devices send ARP 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 ARP Punt Protection feature addresses this situation.

Note Even when the Excessive ARP Punt Protection feature is not enabled, the "bad actors" can affect services for only other devices; they cannot bring down the router.

The Excessive ARP Punt Protection feature is supported on non-subscriber interfaces such as L2 and L3 VLAN sub-interfaces and bundle virtual interfaces (BVIs). 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 default penalty rate, which is non-configurable, is 10 packets per second (pps).

Functioning of Excessive ARP Punt Protection Feature

The Excessive ARP Punt Protection feature monitors ARP control packet traffic arriving from non-subscriber interfaces. These could be physical interfaces, sub-interfaces, or BVIs. It divides interfaces into two categories:

- "Parent" interfaces, which can have other interfaces under them.
- "Non-parent" interfaces, which have no interfaces under them.

A physical interface is always a parent interface because it has VLAN sub-interfaces. An L3 VLAN sub-interface can either be a parent or a non-parent interface.

When a flow is trapped, the Excessive ARP Punt Protection feature tries to identify the source of the flow. The first thing it determines is from which interface the flow came. If this interface is not a "parent" interface, then the feature assumes that it is the end-point source of the flow and penalty policing is applied only on the non-parent interface and not the parent interface. The software applies a penalty-policer in the case of a BVI interface also. If the trapped interface is a "parent" interface, then the entire interface is penalized, which would penalize all the interfaces under it.

For more information about enabling the Excessive ARP Punt Protection feature, see Enabling the Excessive ARP Punt Protection, on page 133.

Note The Excessive ARP Punt Protection feature monitors all punt ARP traffic. You can exclude a particular interface on the router from the monitoring but a remote interface cannot be prevented from being flagged as bad if it is the source of excessive flows.

Bad actors are policed for ARP protocol. There is a static punt rate and a penalty rate for ARP protocol. 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.

Note The excessive rate required to cause an interface to get trapped has nothing to do with the static punt rate (that is,1000 pps). 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.

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 (that is, packet dropping). If there is still an excessive rate of ARP control packet traffic coming from the remote device, then the remote interface is trapped again.

Interface-based Flow

For the Elephant Trap sampler, the MAC address is one of the key fields used to uniquely identify a flow. Certain cases of DoS attacks have dynamically changing source MAC addresses. An individual flow does not cross the threshold in such cases, and hence the Excessive Punt Flow Trap (EPFT) does not trap the flow.

With the interface-based flow feature, Elephant Trap does not consider MAC addresses as a key for uniquely identifying a flow. Hence, all packets received on a non-subscriber interface (irrespective of the source MAC address) are considered to be a part of a single flow. When excessive punts are received on the interface, EPFT does *ifhandle*-based trap, thereby penalty policing the punt traffic on that particular interface.

To enable interface-based flow, use the following command in global configuration mode:

lpts punt excessive-flow-trap interface-based-flow

Note You cannot enable this command if EPFT is turned on for the subscriber-interfaces and non-subscriber-interfaces MAC, or vice versa. This is because interface-based flow feature is mutually exclusive with MAC-based EPFT on non-subscriber interface feature.

Restrictions

These restrictions apply to implementing Excessive ARP Punt Protection feature:

- This feature is non-deterministic. In some cases, the Excessive ARP Punt Protection feature can give a false positive, that is, it could trap an interface that is sending legitimate punt traffic.
- The Excessive ARP Punt Protection 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. The Excessive ARP Punt Protection has safeguards against triggering on short bursts, but longer bursts could trigger a false positive trap.

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* {**default** | **known**} {**rate** *rate*}
4. **commit**
5. **show lpts pifib hardware policer** [**location** {**all** | *node_id*}]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure**	

	Command or Action	Purpose

Step 2	**lpts pifib hardware police** [**location** *node-id*]
	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)#`
	`RP/0/RP0/CPU0:router(config)# lpts pifib hardware police`
	`RP/0/RP0/CPU0:router(config-pifib-policer-global)#`
		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.

Step 3	**flow** *flow_type* {**default** | **known**} {**rate** *rate*}
	Example:
	`RP/0/RP0/CPU0:router(config-pifib-policer-per-node)# flow ospf unicast default rate 20000`
		Configures the policer for the LPTS flow type. The example shows how to configure the policer for the ospf flow type.
		• Use the *flow_type* argument to select the applicable flow type. For information about the flow types, see *IP Addresses and Services Command Reference for Cisco NCS 6000 Series Routers*.
		• Use the **rate** keyword to specify the rate in packets per seconds (PPS). The range is from 0 to 4294967295.
		Note LPTS policy for ntp-default flow type, supports a flow rate of 100 pps on Cisco ASR 9000 Series Router.

Step 4	**commit**	
Step 5	**show lpts pifib hardware policer** [**location** {**all** | *node_id*}] **Example:** `RP/0/RP0/CPU0:router# show lpts pifib hardware policer location 0/2/cpu0`	Displays the policer configuration value set. • (Optional) Use the **location** keyword to display pre-Internal Forwarding Information Base (IFIB) information for the designated node. The *node-id* argument is entered in the *rack/slot/module* notation. • Use the **all** keyword to specify all locations.

Enabling the Excessive ARP Punt Protection

Perform this task to enable the Excessive ARP Punt Protection feature for non-subscriber interfaces. You can also set the penalty timeout and disable the protection on a particular interface in the router.

SUMMARY STEPS

1. **configure**
2. **lpts punt excessive-flow-trap non-subscriber-interfaces**
3. **lpts punt excessive-flow-trap penalty-timeout arp** *time*

4. *(Optional)* **lpts punt excessive-flow-trap exclude** *interface-type interface-id*
5. **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 ARP Punt Protection feature on non-subscriber interfaces.
Step 3	**lpts punt excessive-flow-trap penalty-timeout arp** *time* **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 4	*(Optional)* **lpts punt excessive-flow-trap exclude** *interface-type interface-id* **Example:** `RP/0/RP0/CPU0:router(config)# lpts punt excessive-flow-trap exclude ethernet 0/0/0/1`	Disables the monitoring of ARP punt protection feature on the specified interface.
Step 5	**commit**	

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
```

```
             Node: 0/2/CPU0:
----------------------------------------
flow_type            priority sw_police_id hw_policer_addr avgrate burst static_avgrate
avgrate_type
-------------------- -------- ------------ --------------- ------- ----- --------------
------------
unconfigured-default low      0            580096          500     100   500            2
UDP-default          low      1            580608          500     100   500            2
TCP-default          low      2            581120          500     100   500            2
Mcast-default        low      3            581632          500     100   500            2
Raw-listen           low      4            582144          500     100   500            2
Raw-default          low      5            582656          500     100   500            2
Fragment             low      6            583168          1000    100   1000           2
OSPF-mc-known        high     7            583680          2000    1000  2000           2
ISIS-known           high     8            584192          2000    1000  2000           2
EIGRP                high     9            584704          1500    750   1500           2
RIP                  high     10           585216          1500    750   1500           2
OSPF-mc-default      low      11           585728          1500    1000  1500           2
ISIS-default         low      12           586240          1500    1000  1500           2
BGP-known            high     13           586752          2500    1200  2500           2
BGP-cfg-peer         mdeium   14           587264          100     1000  2000           0
BGP-default          low      15           587776          100     750   1500           1
PIM-mcast-default    mdeium   16           588288          23000   100   23000          2
PIM-ucast            low      17           588800          10000   100   10000          2
IGMP                 mdeium   18           589312          3500    100   3500           2
ICMP-local           mdeium   19           589824          2500    100   2500           2
ICMP-app             low      20           590336          2500    100   2500           2
ICMP-default         low      21           590848          2500    100   2500           2
LDP-TCP-known        mdeium   22           591360          2500    1250  2500           2
LMP-TCP-known        mdeium   23           591872          2500    1250  2500           2
RSVP-UDP             mdeium   24           592384          7000    600   7000           2
RSVP-default         mdeium   25           592896          500     100   500            2
RSVP-known           mdeium   26           593408          7000    600   7000           2
IKE                  mdeium   27           593920          1000    100   1000           2
IPSEC-default        low      28           594432          1000    100   1000           2
IPSEC-known          mdeium   29           594944          3000    100   3000           2
MSDP-known           mdeium   30           595456          1000    100   1000           2
MSDP-cfg-peer        mdeium   31           595968          1000    100   1000           2
MSDP-default         low      32           596480          1000    100   1000           2
SNMP                 low      33           596992          2000    100   2000           2
NTP-default          high     34           597504          500     100   500            2
SSH-known            mdeium   35           598016          1000    100   1000           2
SSH-default          low      36           598528          1000    100   1000           2
HTTP-known           mdeium   37           599040          1000    100   1000           2
HTTP-default         low      38           599552          1000    100   1000           2
SHTTP-known          mdeium   39           600064          1000    100   1000           2
SHTTP-default        low      40           600576          1000    100   1000           2
TELNET-known         mdeium   41           601088          1000    100   1000           2
TELNET-default       low      42           601600          1000    100   1000           2
CSS-known            mdeium   43           602112          1000    100   1000           2
CSS-default          low      44           602624          1000    100   1000           2
RSH-known            mdeium   45           603136          1000    100   1000           2
RSH-default          low      46           603648          1000    100   1000           2
UDP-known            mdeium   47           604160          25000   100   25000          2
TCP-known            mdeium   48           604672          25000   100   25000          2
TCP-listen           low      49           605184          25000   100   25000          2
TCP-cfg-peer         mdeium   50           605696          25000   100   25000          2
Mcast-known          mdeium   51           606208          25000   100   25000          2
LDP-TCP-cfg-peer     mdeium   52           606720          2000    1000  2000           2
LMP-TCP-cfg-peer     mdeium   53           607232          2000    1000  2000           2
LDP-TCP-default      low      54           607744          1500    750   1500           2
LMP-TCP-default      low      55           608256          1500    750   1500           2
UDP-listen           low      56           608768          4000    100   4000           2
UDP-cfg-peer         mdeium   57           609280          4000    100   4000           2
```

```
LDP-UDP              mdeium   58           609792          2000    1000  2000           2
LMP-UDP              mdeium   59           610304          2000    1000  2000           2
All-routers          high     60           610816          1000    500   1000           2
OSPF-uc-known        high     61           611328          2000    1000  2000           2
OSPF-uc-default      low      62           611840          100     100   1000           0
ip-sla               high     63           612352          10000   100   10000          2
ICMP-control         high     64           612864          2500    100   2500           2
L2TPv3               mdeium   65           613376          25000   100   25000          2
PCEP                 mdeium   66           613888          100     200   100            2
GRE                  high     67           614400          1000    1000  1000           2
VRRP                 mdeium   68           614912          1000    1000  1000           2
HSRP                 mdeium   69           615424          400     400   400            2
BFD-known            critical 70           615936          8500    300   8500           2
BFD-default          critical 71           616448          8500    100   8500           2
MPLS-oam             mdeium   72           616960          100     100   100            2
DNS                  mdeium   73           617472          500     100   500            2
RADIUS               mdeium   74           617984          7000    600   7000           2
TACACS               mdeium   75           618496          500     100   500            2
PIM-mcast-known      mdeium   76           619008          23000   100   23000          2
BFD-MP-known         mdeium   77           619520          8400    1024  8400           2
BFD-MP-0             mdeium   78           620032          128     100   128            2
L2TPv2-default       mdeium   79           620544          700     100   700            2
NTP-known            high     80           621056          500     100   500            2
L2TPv2-known         mdeium   81           621568          2000    100   2000           2
```

Enabling Excessive ARP Punt Protection: Example

This example shows the Excessive ARP Punt Protection enabled for non-subscriber interfaces with the ARP penalty timeout set to two minutes and the protection disabled on one of the interfaces on the router.

```
!
configure
lpts punt excessive-flow-trap non-subscriber-interfaces
lpts punt excessive-flow-trap penalty-timeout arp 2
lpts punt excessive-flow-trap exclude interface TenGigE 0/0/0/0
end
!
```

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	*Cisco LPTS Commands* module in the *IP Addresses and Services Command Reference for Cisco NCS 6000 Series Routers*

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 10

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 5.0.0	This feature was introduced.

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.

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.

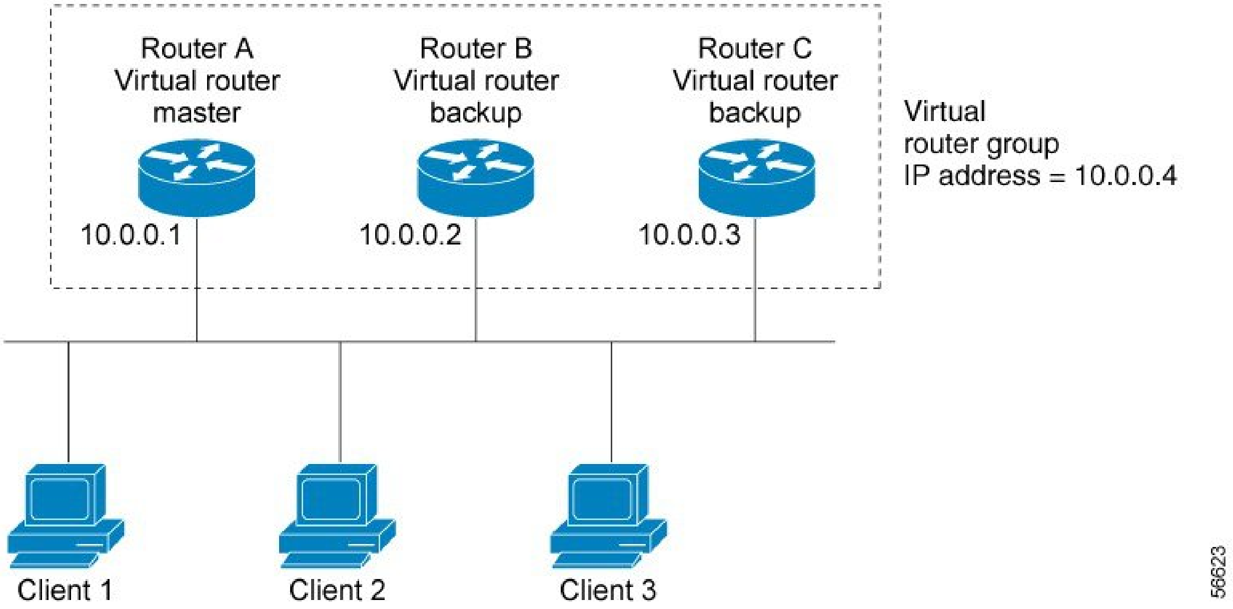
The drawback to dynamic discovery protocols is that they incur some configuration and processing overhead on the LAN client. Also, in the event of a router failure, the process of switching to another router can be slow.

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 15: Basic VRRP Topology, on page 140 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 15: 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 virtual routers (VRRP groups) on a router 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** [**interface***type interface-path-id* [*vrid*]]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**clear vrrp statistics** [**interface***type interface-path-id* [*vrid*]] **Example:** `RP/0/RP0/CPU0:router# clear vrrp statistics`	Clears all software counters for the specified virtual router. • If no interface is specified, statistics of all virtual routers are removed.

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	**configure**	
Step 2	**snmp-server traps vrrp events** **Example:** `RP/0/RP0/CPU0:router(config)snmp-server traps vrrp events`	Enables the SNMP server notifications for VRRP.
Step 3	**commit**	

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:FFFE: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
vrrp 1 priority 120
vrrp 1 text-authentication cisco
vrrp 1 timer 3
vrrp 1 ipv4 10..0.
vrrp 5 timer 30
vrrp 5 ipv4 10..0.
preempt disable
vrrp 100 ipv4 10..0.
commit
```

Router B:

```
config
interface HundredGigE 0/4/0/4
ipv4 address 10.1.0.2/24
exit
router vrrp
interface HundredGigE 0/4/0/4
vrrp 1 priority 100
vrrp 1 text-authentication cisco
vrrp 1 timer 3
vrrp 1 ipv4 10..0.
vrrp 5 priority 200
vrrp 5 timer 30
vrrp 5 ipv4 10..0.
preempt disable
vrrp 100 ipv4 10..0.
commit
```

In the configuration example, each group has the following properties:

- Virtual Router 1:
  - Virtual IP address is 10. .0. .
  - Router A will become the IP address owner router for this group with priority 120.
  - Advertising interval is 3 seconds.
  - is .
  - is enabled.
- Virtual Router 5:
  - is .
  - is .
- Virtual Router 100:
  - Advertising interval is the default 1 second.
  - Preemption is .
  - 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:

```
RP/0/RP0/CPU0:router# show vrrp statistics
show vrrp statistics
Invalid packets:
 Invalid checksum:              0
 Unknown/unsupported versions:  0
 Invalid vrID:                  10
 Too short:                     0
Protocol:
 Transitions to Master          6
Packets:
 Total received:                155
 Bad TTL:                       0
 Failed authentication:         0
 Unknown authentication:        0
 Conflicting authentication:    0
 Unknown Type field:            0
 Conflicting Advertise time:    0
 Conflicting Addresses:         0
 Received with zero priority:   3
 Sent with zero priority:       3


RP/0/RP0/CPU0:router# clear vrrp statistics
RP/0/RP0/CPU0:router
```

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	*Quality of Service Commands on Modular QoS Command Reference for Cisco NCS 6000 Series Routers*
Class-based traffic shaping, traffic policing, low-latency queuing, and Modified Deficit Round Robin (MDRR)	*Configuring Modular Quality of Service Congestion Management on Modular QoSConfiguration Guide for Cisco NCS 6000 Series Routers*
WRED, RED, and tail drop	*Configuring Modular QoS Congestion Avoidance on Modular QoSConfiguration Guide for Cisco NCS 6000 Series Routers*
VRRP commands	*VRRP Commands on IP Addresses and Services Command Reference for Cisco NCS 6000 Series Routers*
Information about user groups and task IDs	*Configuring AAA Services on System Security Configuration Guide for Cisco NCS 6000 Series Routers*

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 11

Configuring Transports

This module provides information about Nonstop Routing (NSR), Transmission Control Protocol (TCP), User Datagram Protocol (UDP), and RAW Transports.

If you have specific requirements and need to adjust the NSR, TCP, UDP, or RAW values, refer to the *Transport Stack Commands on IP Addresses and Services Command Reference for Cisco NCS 6000 Series Routers*.

Feature History for Configuring NSR, TCP, UDP, and UDP RAW Transports

Release	Modification
Release 5.0.0	This feature was introduced.

Prerequisites for Configuring NSR, TCP, UDP, and RAW Transports

The following prerequisites are required to implement NSR, 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, TCP, UDP, and RAW Transports

To configure NSR, TCP, UDP, and RAW transports, you must understand the following concepts:

NSR Overview

Nonstop Routing (NSR) is provided for Open Shortest Path First (OSPF), Border Gateway Protocol (BGP), and Label Distribution Protocol (LDP) protocols for the following events:

- Route Processor (RP) failover
- Process restart for either OSPF, LDP, or TCP
- Online insertion removal (OIR)

In the case of the RP failover, NSR is achieved by for both TCP and the applications (OSPF, BGP, 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 NCS 6000 Series Routers*.

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, 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 Cisco NCS 6000 Series Routers*.

For information on how to configure NSR on a per-process level for each process, refer to the *Routing Configuration Guide for Cisco NCS 6000 Series 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 switchover`	Configures failover as a recovery action for active instances to switch over to a standby route processor (RP) to maintain nonstop routing (NSR.
Step 3	**commit**	

Additional References

The following sections provide references related to configuring NSR, TCP, and transports.

Related Documents

Related Topic	Document Title
Transport Stack commands: complete command syntax, command modes, command history, defaults, usage guidelines, and examples	*Transport Stack Commands in the IP Addresses and Services Command Reference for Cisco NCS 6000 Series Routers*

Related Topic	Document Title
MPLS LDP commands: complete command syntax, command modes, command history, defaults, usage guidelines, and examples	*MPLS Label Distribution Protocol Commands in the MPLS Command Reference for Cisco NCS 6000 Series Routers*
OSPF commands: complete command syntax, command modes, command history, defaults, usage guidelines, and examples	*OSPF Commands in the Routing Command Reference for Cisco NCS 6000 Series Routers*
MPLS Label Distribution Protocol feature information	*Implementing MPLS Label Distribution Protocol in the MPLS Configuration Guide for Cisco NCS 6000 Series Routers*
OSPF feature information	*Implementing OSPF in the Routing Configuration Guide for Cisco NCS 6000 Series Routers*

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