IPv6 Configuration Guide, Cisco IOS XE Release 3S

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

CONTENTS

Implementing IPv6 Addressing and Basic Connectivity

Implementing basic IPv6 connectivity in the Cisco IOS software consists of assigning IPv6 addresses to individual router interfaces. The forwarding of IPv6 traffic can be enabled globally, and Cisco Express Forwarding switching for IPv6 can also be enabled. Basic connectivity can be enhanced by configuring support for AAAA record types in the Domain Name System (DNS) name-to-address and address-to-name lookup processes, and by managing IPv6 neighbor discovery.

Finding Feature Information

Your software release may not support all the features documented in this module. For the latest caveats and feature information, see Bug Search Tool and the release notes for your platform and software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the feature information table at the end of this module.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Prerequisites for Implementing IPv6 Addressing and Basic Connectivity

- The following prerequisites apply to Cisco Express Forwarding and distributed Cisco Express Forwarding for IPv6:
    - To forward IPv6 traffic using Cisco Express Forwarding or distributed Cisco Express Forwarding, you must configure forwarding of IPv6 unicast datagrams globally on the router by using the **ipv6**

unicast-routing command, and you must configure an IPv6 address on an interface by using the **ipv6 address** command.

- You must enable Cisco Express Forwarding for IPv4 globally on the router by using the **ip cef** command before enabling Cisco Express Forwarding for IPv6 globally on the router by using the **ipv6 cef** command.
- On distributed architecture platforms that support both Cisco Express Forwarding and distributed Cisco Express Forwarding, you must enable distributed Cisco Express Forwarding for IPv4 globally on the router by using the **ip cef distributed** command before enabling distributed Cisco Express Forwarding for IPv6 globally on the router by using the **ipv6 cef distributed** command.
- To use Unicast Reverse Path Forwarding (RPF), enable Cisco Express Forwarding switching or distributed Cisco Express Forwarding switching in the router. There is no need to configure the input interface for Cisco Express Forwarding switching. As long as Cisco Express Forwarding is running on the router, individual interfaces can be configured with other switching modes.

Note For Unicast RPF to work, Cisco Express Forwarding must be configured globally in the router. Unicast RPF will not work without Cisco Express Forwarding.

Restrictions for Implementing IPv6 Addressing and Basic Connectivity

- In Cisco IOS Release 12.2(11)T or earlier releases, IPv6 supports only process switching for packet forwarding. Cisco Express Forwarding switching and distributed Cisco Express Forwarding switching for IPv6 are supported in Cisco IOS Release 12.2(13)T. Distributed Cisco Express Forwarding switching for IPv6 is supported in Cisco IOS Release 12.0(21)ST.
- IPv6 packets are transparent to Layer 2 LAN switches because the switches do not examine Layer 3 packet information before forwarding IPv6 frames. Therefore, IPv6 hosts can be directly attached to Layer 2 LAN switches.
- In any Cisco IOS release with IPv6 support, multiple IPv6 global addresses within the same prefix can be configured on an interface. However, multiple IPv6 link-local addresses on an interface are not supported. See the "Mapping IPv6 Addresses to IPv6 ATM and Frame Relay Interfaces" section for information on configuring multiple IPv6 global addresses within the same prefix on an interface.
- Because RFC 3879 deprecates the use of site-local addresses, configuration of private IPv6 addresses should be done following the recommendations of unique local addressing (ULA) in RFC 4193.
- Bridge-Group Virtual Interfaces (BVIs) in IPv6 are not supported with NAT-PT and wireless interfaces Dot11Radio.

Information About Implementing IPv6 Addressing and Basic Connectivity

- IPv6 for Cisco Software, page 3
- Large IPv6 Address Space for Unique Addresses, page 3
- IPv6 Address Formats, page 4

IPv6 for Cisco 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 the 32-bit addressing scheme of IP version 4 (IPv4) proved to be inadequate to meet the demands of Internet growth. IPv6 is based on IP but with a much larger address space and improvements such as a simplified main header and extension headers. IPv6 is described in RFC 2460, *Internet Protocol, Version 6 (IPv6) Specification*, issued by the Internet Engineering Task Force (IETF). Further RFCs describe the architecture and services supported by IPv6.

The architecture of IPv6 has been designed to allow existing IPv4 users to 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 Routing Information Protocol (RIP), Integrated Intermediate System-to-Intermediate System (IS-IS), Open Shortest Path First version 3 (OSPFv3), and multiprotocol Border Gateway Protocol (BGP). Other available features include stateless autoconfiguration, enhanced support for Mobile IPv6, and an increased number of multicast addresses.

Large IPv6 Address Space for Unique Addresses

The primary motivation for IPv6 is the need to meet 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; therefore, IPv6 enables new application protocols that do not require special processing by border devices 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:DB8:7654:3210:FEDC:BA98:7654:3210

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

IPv6 addresses commonly contain successive hexadecimal fields of zeros. Two colons (::) may 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). The table below 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 1 Compressed IPv6 Address Formats

IPv6 Address Type	Preferred Format	Compressed Format
Unicast	2001:0:0:0:DB8:800:200C:417A	2001::DB8:800:200C: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 the table above 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 devices do not forward packets that have the IPv6 loopback address as their source or destination address.

The unspecified address listed in the table above 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* must be in the form documented in RFC 2373 where 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 comprise the prefix (the network portion of the address). For example, 2001:DB8:8086:6502::/32 is a valid IPv6 prefix.

IPv6 Address Type: Unicast

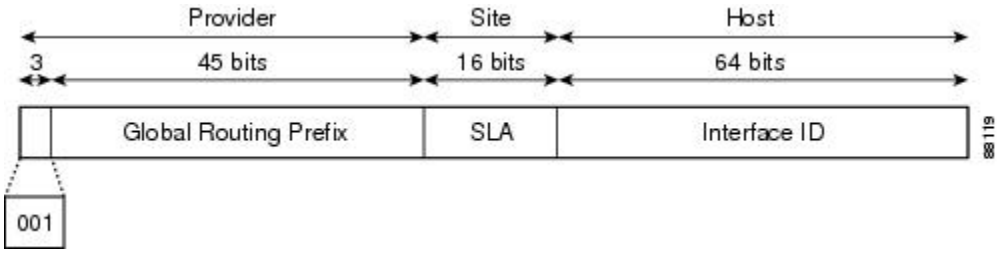
An IPv6 unicast address is an identifier for a single interface, on a single node. A packet that is sent to a unicast address is delivered to the interface identified by that address. Cisco software supports the IPv6 unicast address types described in the following sections.

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 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 IPv6 global unicast address allocation uses the range of addresses that start with binary value 001 (2000::/3). The figure below shows the structure of an aggregatable global address.

Figure 1 Aggregatable Global Address Format



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

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

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

An interface ID is used to identify interfaces on a link. The interface ID must be unique to the link. It may also be unique over a broader scope. In many cases, an interface ID will be 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 and FDDI interfaces), the first three octets (24 bits) are taken from the Organizationally Unique Identifier (OUI) of the 48-bit link-layer address (the media access control, or 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 other interface types (for example, ATM, Frame Relay, loopback, serial, and tunnel interface types except tunnel interfaces used with IPv6 overlay tunnels), the interface ID is constructed in the same way as the interface ID for IEEE 802 interface types; however, the first MAC address from the pool of MAC addresses in the device is used to construct the identifier (because the interface does not have a MAC address).
- For tunnel interface types that are used with IPv6 overlay tunnels, the interface ID is the IPv4 address assigned to the tunnel interface with all zeros in the high-order 32 bits of the identifier.

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

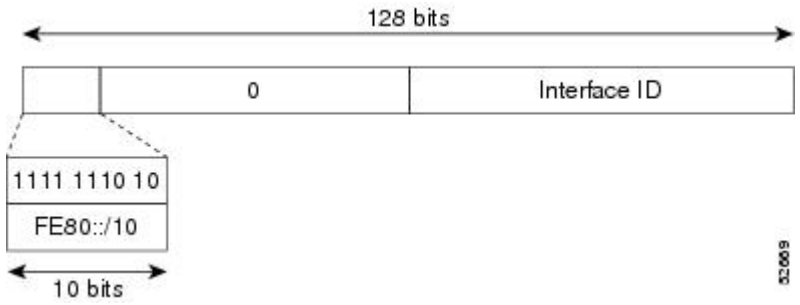
If no IEEE 802 interface types are in the device, link-local IPv6 addresses are generated on the interfaces in the device in the following sequence:

1. The device is queried for MAC addresses (from the pool of MAC addresses in the device).
2. If no MAC addresses are available in the device, the serial number of the device is used to form the link-local addresses.
3. If the serial number of the device cannot be used to form the link-local addresses, the device uses a message digest algorithm 5 (MD5) hash to determine the MAC address of the device from the hostname of the device.

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 globally unique addresses to communicate. The figure below shows the structure of a link-local address.
IPv6 devices must not forward packets that have link-local source or destination addresses to other links.

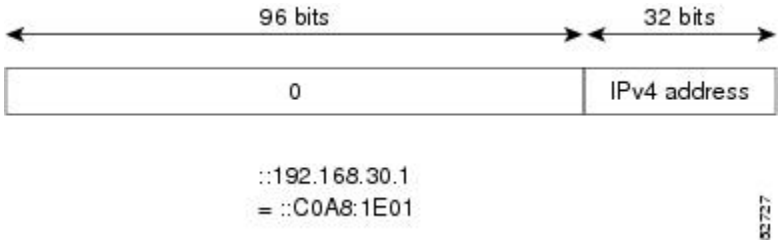
Figure 2 Link-Local Address Format



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. The figure below shows the structure of an IPv4-compatible IPv6 address and a few acceptable formats for the address.

Figure 3 IPv4-Compatible IPv6 Address Format



Unique Local Address

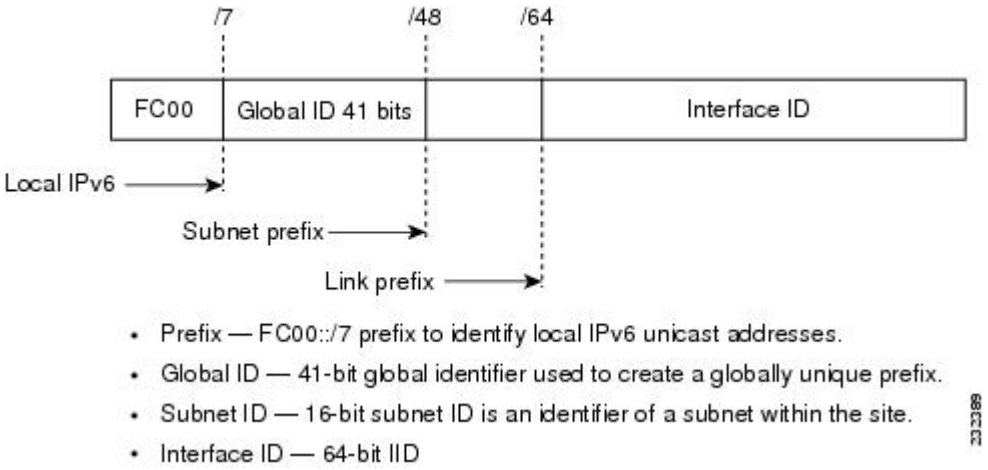
A unique local address is an IPv6 unicast address that is globally unique and is intended for local communications. It is not expected to be routable on the global Internet and is routable inside of a limited area, such as a site. It may also be routed between a limited set of sites.

A unique local address has the following characteristics:

- It has a globally unique prefix (that is, it has a high probability of uniqueness).
- It has a well-known prefix to allow for easy filtering at site boundaries.
- It allows sites to be combined or privately interconnected without creating any address conflicts or requiring renumbering of interfaces that use these prefixes.
- It is ISP-independent and can be used for communications inside of a site without having any permanent or intermittent Internet connectivity.
- If it is accidentally leaked outside of a site via routing or DNS, there is no conflict with any other addresses.
- Applications may treat unique local addresses like global scoped addresses.

The figure below shows the structure of a unique local address.

Figure 4 Unique Local Address Structure



Site-Local Address

Because RFC 3879 obsoletes the use of site-local addresses, configuration of private IPv6 addresses should be done following the recommendations of unique local addressing in RFC 4193.

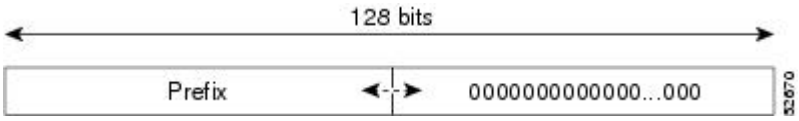
IPv6 Address Type: Anycast

An anycast address is an address that is assigned to a set of interfaces that typically belong to different nodes. A packet sent to an anycast address is delivered to the closest interface (as defined by the routing protocols in use) identified by the anycast address. Anycast addresses are syntactically indistinguishable from unicast addresses, because anycast addresses are allocated from the unicast address space. Assigning a unicast address to more than one interface makes a unicast address an anycast address. Nodes to which the anycast address is assigned must be explicitly configured to recognize that the address is an anycast address.

Note Anycast addresses can be used only by a device, not a host, and anycast addresses must not be used as the source address of an IPv6 packet.

The figure below shows the format of the subnet device anycast address; the address has a prefix concatenated by a series of zeros (the interface ID). The subnet device anycast address can be used to reach a device on the link that is identified by the prefix in the subnet device anycast address.

Figure 5 Subnet Device Anycast Address Format

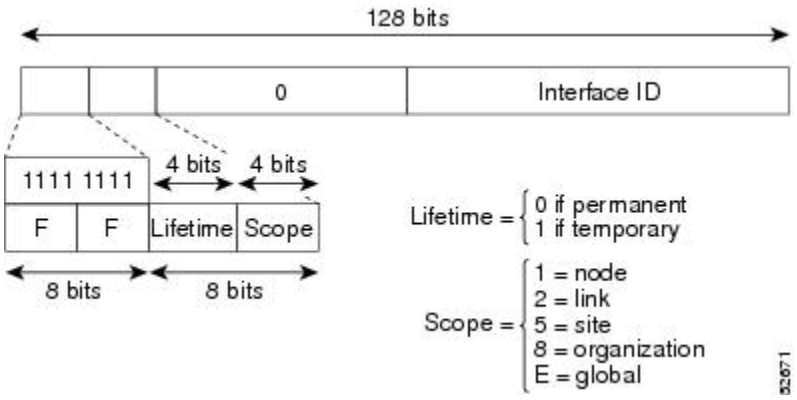


IPv6 Address Type Multicast

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. The figure below shows the format of the IPv6 multicast address.

Figure 6 IPv6 Multicast Address Format



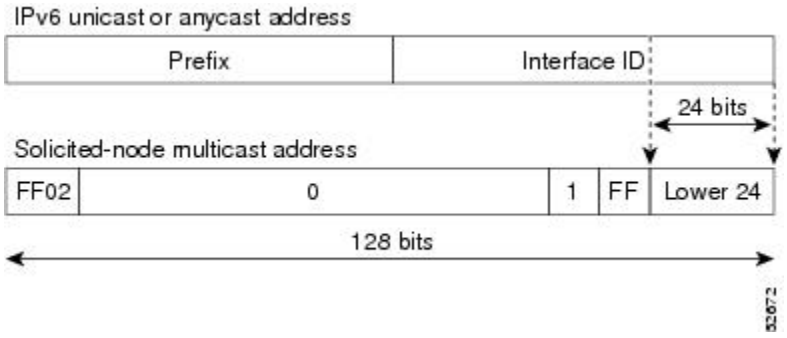
IPv6 nodes (hosts and routers) are required to join (receive packets destined for) the following multicast groups:

- 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 or anycast address (see the figure below). 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 7 IPv6 Solicited-Node Multicast Address Format



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

- IPv6 Multicast Groups, page 10

IPv6 Multicast Groups

An IPv6 address must be configured on an interface before the interface can forward IPv6 traffic. Configuring a site-local or 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 and anycast address assigned to the interface

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

- All-nodes link-local multicast group FF02::1
- All-routers link-local multicast group FF02::2

IPv6 Address Output Display

When IPv6 or IPv4 command output displays an IPv6 address, a long IPv6 address can overflow into neighboring fields, causing the output to be difficult to read. The output fields were designed to work with the longest possible IPv4 address, which has 15 characters; IPv6 addresses can be up to 39 characters long. The following scheme has been adopted in IPv4 and IPv6 commands to allow the appropriate length of IPv6 address to be displayed and move the following fields to the next line, if necessary. The fields that are moved are kept in alignment with the header row.

The following example displays eight connections. The first six connections feature IPv6 addresses; the last two connections feature IPv4 addresses.

```
Device# where
Conn Host                Address             Byte  Idle Conn Name
   1 test5               2001:DB8:3333:4::5    6    24 test5
   2 test4               2001:DB8:3333:44::5
                                               6    24 test4
   3 2001:DB8:3333:4::5 2001:DB8:3333:4::5     6    24 2001:DB8:3333:4::5
   4 2001:DB8:3333:44::5
                         2001:DB8:3333:44::5
                                               6    23 2001:DB8:3333:44::5
   5 2001:DB8:3000:4000:5000:6000:7000:8001
                         2001:DB8:3000:4000:5000:6000:7000:8001
                                               6    20 2001:DB8:3000:4000:5000:6000:
   6 2001:DB8:1::1      2001:DB8:1::1          0     1 2001:DB8:1::1
   7 10.1.9.1            10.1.9.1              0     0 10.1.9.1
   8 10.222.111.222      10.222.111.222        0     0 10.222.111.222
```

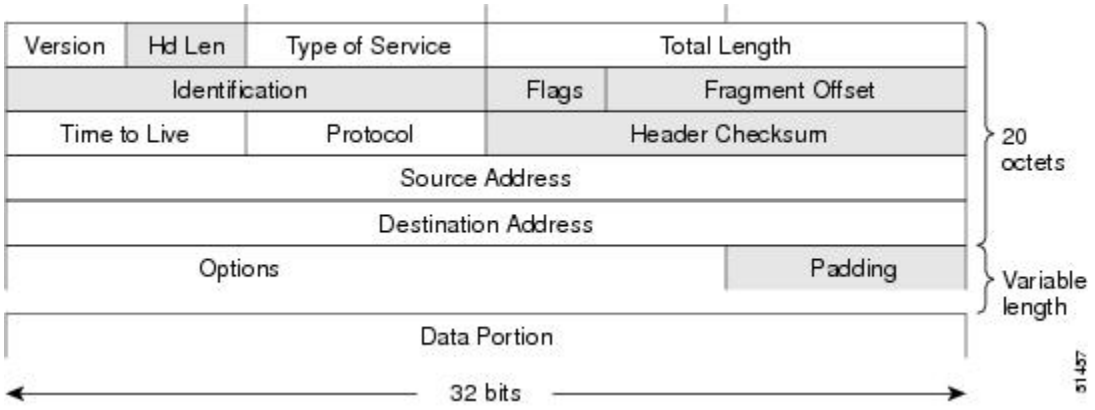
Connection 1 contains an IPv6 address that uses the maximum address length in the address field. Connection 2 shows the IPv6 address overflowing the address field and the following fields moved to the next line, but in alignment with the appropriate headers. Connection 3 contains an IPv6 address that fills the maximum length of the hostname and address fields without wrapping any lines. Connection 4 shows the effect of both the hostname and address fields containing a long IPv6 address. The output is shown over three lines keeping the correct heading alignment. Connection 5 displays a similar effect as connection 4 with a very long IPv6 address in the hostname and address fields. Note that the connection name field is actually truncated. Connection 6 displays a very short IPv6 address that does not require any change in the display. Connections 7 and 8 display short and long IPv4 addresses.

Note The IPv6 address output display applies to all commands that display IPv6 addresses.

Simplified IPv6 Packet Header

The basic IPv4 packet header has 12 fields with a total size of 20 octets (160 bits) (see the figure below). 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 shown in the figure below are not included in the IPv6 packet header.

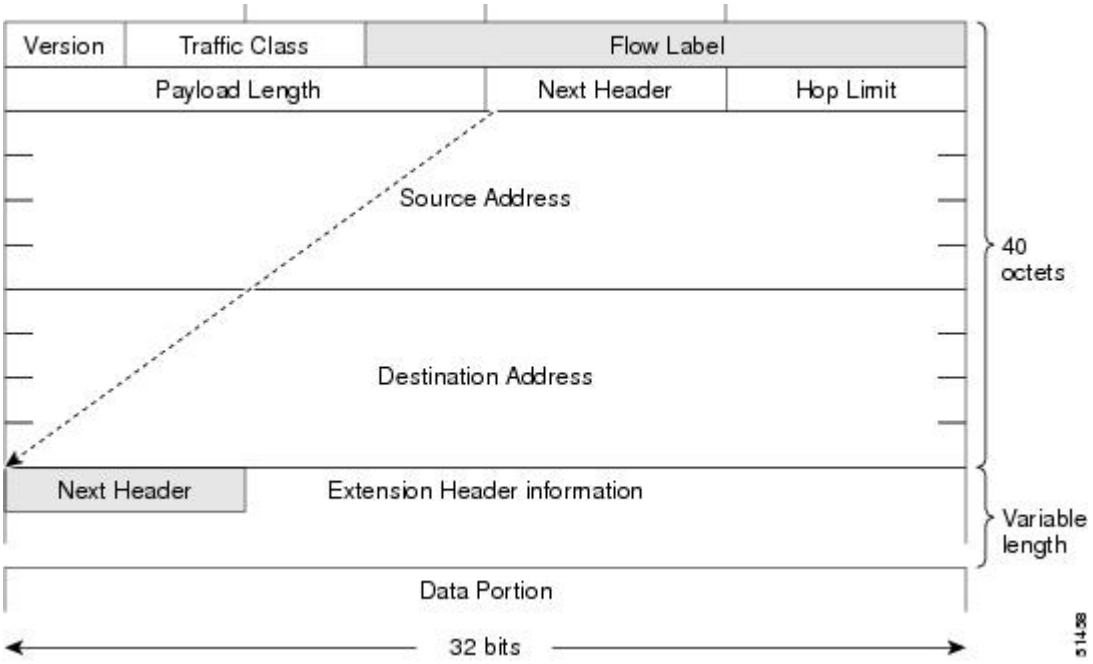
Figure 8 IPv4 Packet Header Format



The basic IPv6 packet header has 8 fields with a total size of 40 octets (320 bits) (see the figure below). Fields were removed from the IPv6 header because, in IPv6, fragmentation is not handled by devices 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 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 9 IPv6 Packet Header Format



The table below lists the fields in the basic IPv6 packet header.

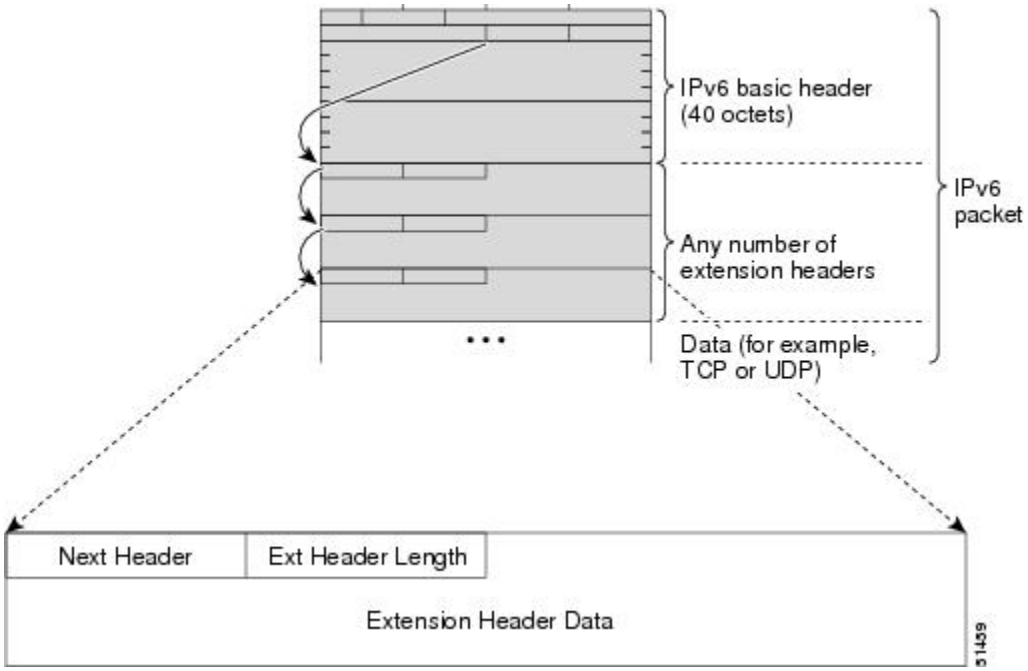
Table 2 Basic IPv6 Packet Header Fields

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.

Field	Description
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 the figure immediately above.
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 devices that an IPv6 packet can pass through before the packet is considered invalid. Each device decrements the value by one. Because no checksum is in the IPv6 header, the device can decrement the value without needing to recalculate the checksum, which saves processing resources.
Source Address	Similar to the Source Address field in the IPv4 packet header, except that the field contains a 128-bit source address for IPv6 instead of a 32-bit source address for IPv4.
Destination Address	Similar to the Destination Address field in the IPv4 packet header, except that the field contains a 128-bit destination address for IPv6 instead of a 32-bit destination address for IPv4.

Following the eight fields of the basic IPv6 packet header are optional extension headers and the data portion of the packet. If present, each extension header is aligned to 64 bits. There is no fixed number of extension headers in an IPv6 packet. 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. The figure below shows the IPv6 extension header format.

Figure 10 IPv6 Extension Header Format



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

Table 3 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.
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 headers	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 headers	135	Extension headers used by mobile nodes, correspondent nodes, and home agents in all messaging related to the creation and management of bindings.

Cisco Express Forwarding and Distributed Cisco Express Forwarding Switching for IPv6

Cisco Express Forwarding is advanced, Layer 3 IP switching technology for the forwarding of IPv6 packets. Distributed Cisco Express Forwarding performs the same functions as Cisco Express Forwarding but for distributed architecture platforms. Distributed Cisco Express Forwarding for IPv6 and Cisco

Express Forwarding for IPv6 function the same and offer the same benefits as for distributed Cisco Express Forwarding for IPv4 and Cisco Express Forwarding for IPv4. Both have network entries that are added, removed, or modified in the IPv6 Routing Information Base (RIB) (as dictated by the routing protocols in use) and are reflected in the Forwarding Information Bases (FIBs), and the IPv6 adjacency tables maintain Layer 2 next-hop addresses for all entries in each FIB.

Each IPv6 device interface has an association to one IPv6 global FIB and one IPv6 link-local FIB (multiple interfaces can have an association to the same FIB). All IPv6 device interfaces that are attached to the same IPv6 link share the same IPv6 link-local FIB. IPv6 packets that have an IPv6 global destination address are processed by the IPv6 global FIB; however, packets that have an IPv6 global destination address and an IPv6 link-local source address are sent to the Route Processor (RP) for process switching and scope-error handling. Packets that have a link-local source address are not forwarded off of the local link and are sent to the RP for process switching and scope-error handling.

- Unicast Reverse Path Forwarding, page 15

Unicast Reverse Path Forwarding

Use the Unicast Reverse Path Forwarding for IPv6 feature to mitigate problems caused by malformed or spoofed IPv6 source addresses that pass through an IPv6 device. Malformed or forged source addresses can indicate denial-of-service (DoS) attacks based on source IPv6 address spoofing.

When uRPF is enabled on an interface, the device examines all packets received on that interface. The device verifies that the source address appears in the routing table and matches the interface on which the packet was received. This "look backward" ability is available only when Cisco Express Forwarding is enabled on the device, because the lookup relies on the presence of the Forwarding Information Bases (FIBs). Cisco Express Forwarding generates the FIB as part of its operation.

Note uRPF is an input function and is applied only on the input interface of a device at the upstream end of a connection.

The uRPF feature verifies whether any packet received at a device interface arrives on one of the best return paths to the source of the packet. The feature performs a reverse lookup in the Cisco Express Forwarding table. If uRPF does not find a reverse path for the packet, uRPF can drop or forward the packet, depending on whether an access control list (ACL) is specified. If an ACL is specified, then when (and only when) a packet fails the uRPF check, the ACL is checked to verify if the packet should be dropped (using a deny statement in the ACL) or forwarded (using a permit statement in the ACL). Regardless of whether a packet is dropped or forwarded, the packet is counted in the global IP traffic statistics for uRPF drops and in the interface statistics for uRPF.

If no ACL is specified, the device drops the forged or malformed packet immediately and no ACL logging occurs. The device and interface uRPF counters are updated.

uRPF events can be logged by specifying the logging option for the ACL entries. Log information can be used to gather information about the attack, such as source address and time.

Note With uRPF, all equal-cost "best" return paths are considered valid. uRPF works in cases where multiple return paths exist, provided that each path is equal to the others in terms of the routing cost (number of hops, weights, and so on) and as long as the route is in the FIB.

DNS for IPv6

IPv6 supports DNS record types that are supported in the DNS name-to-address and address-to-name lookup processes. The DNS record types support IPv6 addresses. IPv6 also supports the reverse mapping of IPv6 addresses to DNS names.

A name server is used to track information associated with domain names. A name server can maintain a database of hostname-to-address mappings. Each name can map to one or more IPv4 addresses, IPv6 addresses, or both address types. In order to use this service to map domain names to IPv6 addresses, you must specify a name server and enable the DNS.

Cisco software maintains a cache of hostname-to-address mappings for use by the **connect**, **telnet**, and **ping** commands, related Telnet support operations, and many other commands that generate command output. This cache speeds the conversion of names to addresses.

Similar to IPv4, IPv6 uses a naming scheme that allows a network device to be identified by its location within a hierarchical name space that provides for domains. Domain names are joined with periods (.) as the delimiting characters. For example, Cisco is a commercial organization that is identified by a com domain name, so its domain name is cisco.com. A specific device in this domain, the FTP server, for example, is identified as ftp.cisco.com.

The following table lists the IPv6 DNS record types.

Table 4 IPv6 DNS Record Types

Record Type	Description	Format
AAAA	Maps a hostname to an IPv6 address. (Equivalent to an A record in IPv4.)	www.abc.test AAAA 3FFE:YYYY:C18:1::2
PTR	Maps an IPv6 address to a hostname. (Equivalent to a pointer record [PTR] in IPv4.) **Note** Cisco software supports resolution of PTR records for the IP6.INT domain.	2.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.1.0.0.0.8.1.c.0.y.y.y.y.e.f.f.3.ip6.int PTR www.abc.test

Path MTU Discovery for IPv6

As in IPv4, path MTU discovery in IPv6 allows a host to dynamically discover and adjust to differences in the MTU size of every link along a given data path. In IPv6, however, fragmentation is handled by the source of a packet when the path MTU of one link along a given data path is not large enough to accommodate the size of the packets. Having IPv6 hosts handle packet fragmentation saves IPv6 router processing resources and helps IPv6 networks run more efficiently.

Note In IPv6, the minimum link MTU is 1280 octets. Cisco recommends using an MTU value of 1500 octets for IPv6 links.

Cisco Discovery Protocol IPv6 Address Support

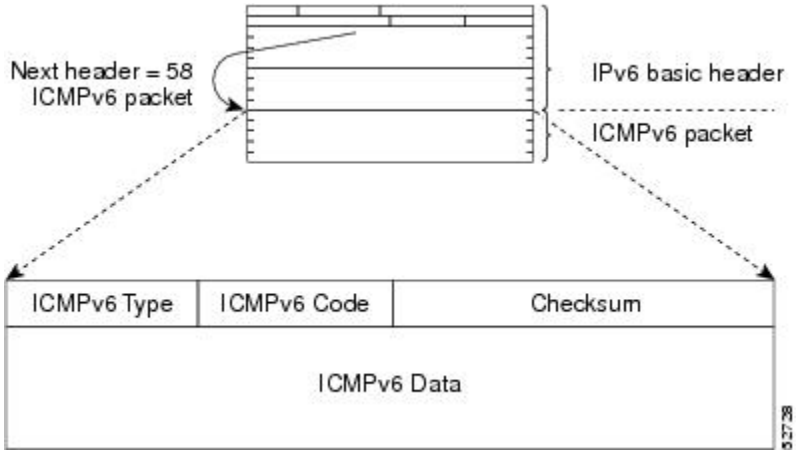
The Cisco Discovery Protocol IPv6 address support for neighbor information feature adds the ability to transfer IPv6 addressing information between two Cisco devices. Cisco Discovery Protocol support for IPv6 addresses provides IPv6 information to network management products and troubleshooting tools.

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, such as 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 devices 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. The figure below shows the IPv6 ICMP packet header format.

Figure 11 IPv6 ICMP Packet Header Format



- IPv6 ICMP Rate Limiting, page 17

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, no IPv6 ICMP error messages are 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.

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 track neighboring devices.

The IPv6 static cache entry for neighbor discovery feature allows static entries to be made in the IPv6 neighbor cache. Static routing requires an administrator to manually enter IPv6 addresses, subnet masks, gateways, and corresponding Media Access Control (MAC) addresses for each interface of each device into a table. Static routing enables more control but requires more work to maintain the table. The table must be updated each time routes are added or changed.

Stateful Switchover

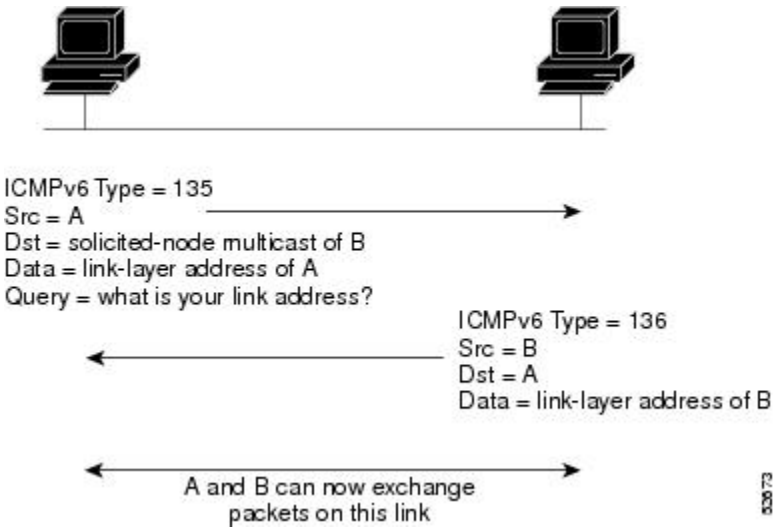
IPv6 neighbor discovery supports stateful switchover (SSO) using Cisco Express Forwarding. When switchover occurs, the Cisco Express Forwarding adjacency state, which is checkpointed, is used to reconstruct the neighbor discovery cache.

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 the figure below). 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 12 IPv6 Neighbor Discovery: Neighbor Solicitation Message



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

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

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

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

Neighbor solicitation messages are also used to verify the reachability of a neighbor after the link-layer address of a neighbor is identified. Neighbor unreachability detection identifies the failure of a neighbor or the failure of the forward path to the neighbor, and is used for all paths between hosts and neighboring nodes (hosts or devices). 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 the receipt of a neighbor advertisement message in response to a neighbor solicitation message. 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 device 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 verified 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 software does not verify the uniqueness of anycast or global addresses that are generated from 64-bit interface identifiers.

Enhanced IPv6 Neighbor Discovery Cache Management

The enhanced IPv6 neighbor discovery cache management feature optimizes IPv6 neighbor discovery by providing ND cache autorefresh, unsolicited neighbor advertisement (NA) gleaning, and neighbor unreachability detection (NUD) exponential retransmit.

The neighbor discovery protocol enforces NUD, which can detect failing nodes or devices and changes to link-layer addresses. NUD is used to maintain reachability information for all paths between hosts and neighboring nodes, including host-to-host, host-to-device, and device-to-host communication.

The neighbor cache maintains mapping information about the IPv6 link-local or global address to the link-layer address. The neighbor cache also maintains the neighbor's reachability state, which is updated using NUD. Neighbors can be in one of the following five possible states:

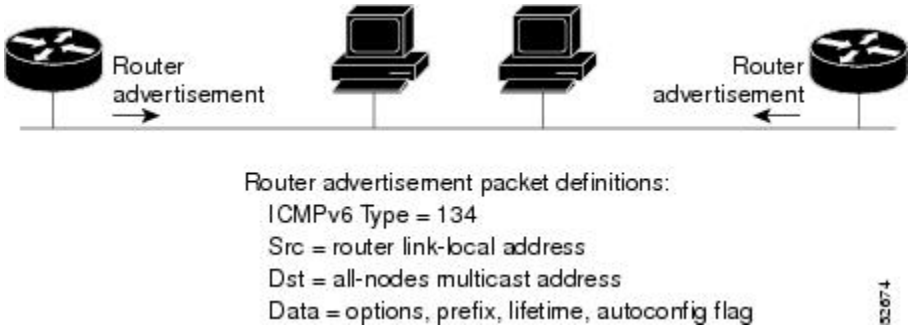
- DELAY—Neighbor is pending re-resolution, and traffic might flow to this neighbor.
- INCOMPLETE—Address resolution is in progress, and the link-layer address is not yet known.
- PROBE—Neighbor re-resolution is in progress, and traffic might flow to this neighbor.
- REACHABLE—Neighbor is known to be reachable within the last reachable time interval.
- STALE—Neighbor requires re-resolution, and traffic may flow to this neighbor.
-

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 device. For stateless autoconfiguration to work properly, the advertised prefix length in RA messages must always be 64 bits.

The RA messages are sent to the all-nodes multicast address (see the figure below).

Figure 13 IPv6 Neighbor Discovery: RA Message



RA 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 stateful) that can be completed
- Default device information (whether the device sending the advertisement should be used as a default device and, if so, the amount of time, in seconds, the device should be used as a default device)
- Additional information for hosts, such as the hop limit and maximum transmission unit (MTU) a host should use in packets that it originates

RAs are also sent in response to device solicitation messages. Device solicitation messages, which have a value of 133 in the Type field of the ICMP packet header, are sent by hosts at system startup so that the host can immediately autoconfigure without needing to wait for the next scheduled RA message. Given that device solicitation messages are usually sent by hosts at system startup (the host does not have a configured unicast address), the source address in device 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 device solicitation message is used as the source address in the message. The destination address in device solicitation messages is the all-devices multicast address with a scope of the link. When an RA is sent in response to a device solicitation, the destination address in the RA message is the unicast address of the source of the device solicitation message.

The following RA message parameters can be configured:

- The time interval between periodic RA messages
- The “device lifetime” value, which indicates the usefulness of a device as the default device (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 RA messages (with default values) is automatically enabled on Ethernet and FDDI interfaces when the **ipv6 unicast-routing** command is

configured. For other interface types, the sending of RA messages must be manually configured by using the **no ipv6 nd ra suppress** command. The sending of RA messages can be disabled on individual interfaces by using the **ipv6 nd ra suppress** command.

- Default Router Preferences for Traffic Engineering, page 22

Default Router Preferences for Traffic Engineering

Hosts discover and select default devices by listening to Router Advertisements (RAs). Typical default device selection mechanisms are suboptimal in certain cases, such as when traffic engineering is needed. For example, two devices on a link may provide equivalent but not equal-cost routing, and policy may dictate that one of the devices is preferred. Some examples are as follows:

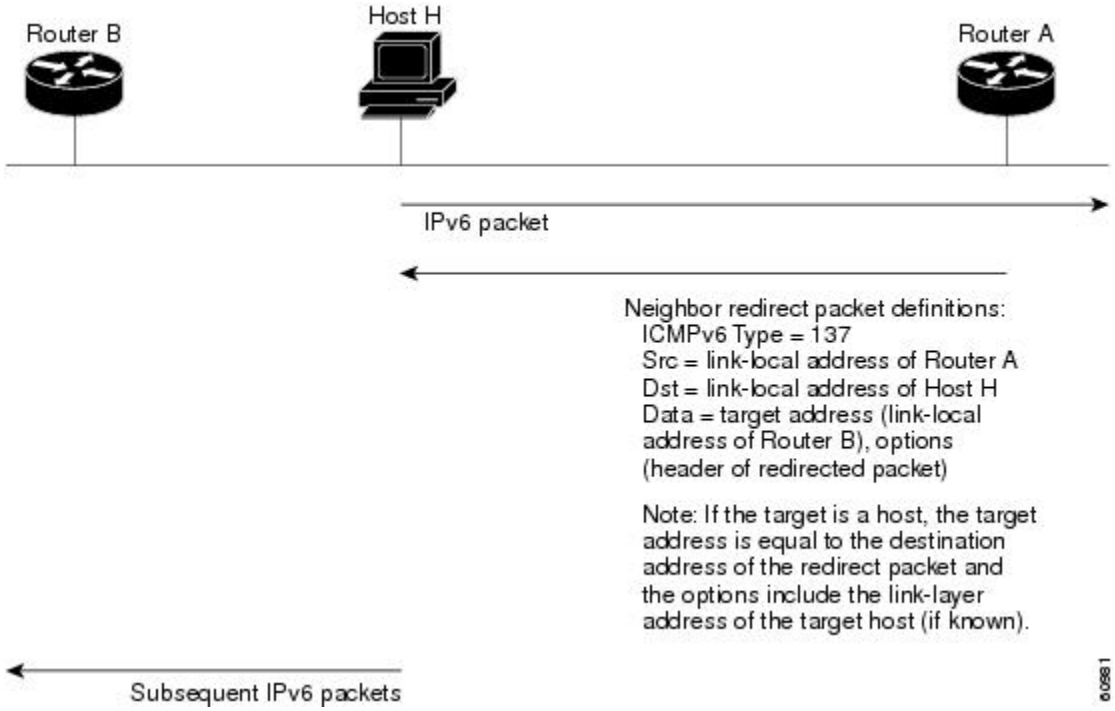
- Multiple devices that route to distinct sets of prefixes—Redirects (sent by nonoptimal devices for a destination) mean that hosts can choose any device and the system will work. However, traffic patterns may mean that choosing one of the devices would lead to considerably fewer redirects.
- Accidentally deploying a new device—Deploying a new device before it has been fully configured could lead to hosts adopting the new device as a default device and traffic disappearing. Network managers may want to indicate that some devices are more preferred than others.
- Multihomed situations—Multihomed situations may become more common, because of multiple physical links and because of the use of tunneling for IPv6 transport. Some of the devices may not provide full default routing because they route only to the 6-to-4 prefix or they route only to a corporate intranet. These situations cannot be resolved with redirects, which operate only over a single link.

The default router preference (DRP) feature provides a basic preference metric (low, medium, or high) for default devices. The DRP of a default device is signaled in unused bits in RA messages. This extension is backward compatible, both for devices (setting the DRP bits) and hosts (interpreting the DRP bits). These bits are ignored by hosts that do not implement the DRP extension. Similarly, the values sent by devices that do not implement the DRP extension will be interpreted by hosts that do implement it as indicating a “medium” preference. DRPs need to be configured manually.

IPv6 Neighbor Redirect Message

A value of 137 in the type field of the ICMP packet header identifies an IPv6 neighbor redirect message. Devices send neighbor redirect messages to inform hosts of better first-hop nodes on the path to a destination (see the figure below).

Figure 14 IPv6 Neighbor Discovery: Neighbor Redirect Message



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

After forwarding a packet, a device 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 device.
- The packet is about to be sent out the interface on which it was received.
- The device 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** command to limit the rate at which the device generates all IPv6 ICMP error messages, including neighbor redirect messages, which ultimately reduces link-layer congestion.

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

Per-Interface Neighbor Discovery Cache Limit

The number of entries in the Neighbor Discovery cache can be limited by interface. Once the limit is reached, no new entries are allowed. The per-interface Neighbor Discovery cache limit function can be used to prevent any particular customer attached to an interface from overloading the Neighbor Discovery cache, whether intentionally or unintentionally.

When this feature is enabled globally, a common per-interface cache size limit is configured on all interfaces on the device. When this feature is enabled per interface, a cache size limit is configured on the associated interface. The per-interface limit overrides any globally configured limit.

Link, Subnet, and Site Addressing Changes

This section describes the IPv6 stateless autoconfiguration and general prefix features, which can be used to manage link, subnet, and site addressing changes.

IPv6 Stateless Autoconfiguration

All interfaces on IPv6 nodes must have a link-local address, which is usually automatically configured from the identifier for an interface and the link-local prefix FE80::/10. A link-local address enables a node to communicate with other nodes on the link and can be used to further configure the node.

Nodes can connect to a network and automatically generate global IPv6 addresses without the need for manual configuration or help of a server, such as a Dynamic Host Configuration Protocol (DHCP) server. With IPv6, a device on the link advertises any global prefixes in Router Advertisement (RA) messages, as well as its willingness to function as a default device for the link. RA messages are sent periodically and in response to device solicitation messages, which are sent by hosts at system startup.

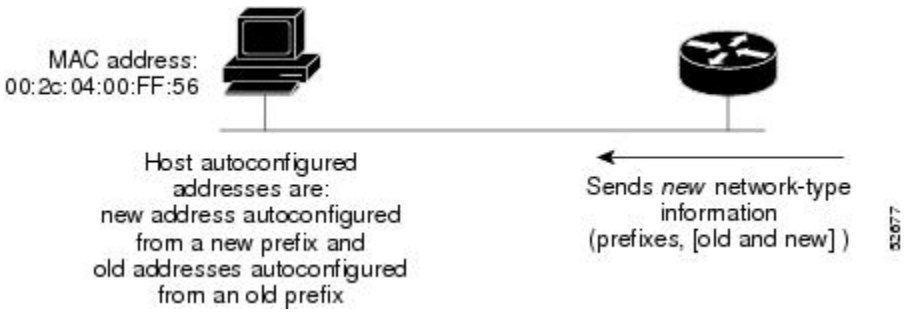
A node on the link can automatically configure global IPv6 addresses by appending its interface identifier (64 bits) to the prefixes (64 bits) included in the RA messages. The resulting 128-bit IPv6 addresses configured by the node are then subjected to duplicate address detection to ensure their uniqueness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

Simplified Network Renumbering for IPv6 Hosts

The strict aggregation of the global routing table requires that networks be renumbered when the service provider for the network is changed. When the stateless autoconfiguration functionality in IPv6 is used to renumber a network, the prefix from a new service provider is added to RA messages that are sent on the link. (The RA messages contain both the prefix from the old service provider and the prefix from the new

service provider.) Nodes on the link automatically configure additional addresses by using the prefix from the new service provider. The nodes can then use the addresses created from the new prefix and the existing addresses created from the old prefix on the link. Configuration of the lifetime parameters associated with the old and new prefixes means that nodes on the link can make the transition to using only addresses created from the new prefix. During a transition period, the old prefix is removed from RA messages and only addresses that contain the new prefix are used on the link (the renumbering is complete) (see the figure below).

Figure 15 IPv6 Network Renumbering for Hosts Using Stateless Autoconfiguration



IPv6 General Prefixes

The upper 64 bits of an IPv6 address are composed from a global routing prefix plus a subnet ID, as defined in RFC 3513. A general prefix (for example, /48) holds a short prefix, based on which a number of longer, more-specific prefixes (for example, /64) can be defined. When the general prefix is changed, all of the more-specific prefixes based on it will change, too. This function greatly simplifies network renumbering and allows for automated prefix definition.

For example, a general prefix might be 48 bits long (“/48”) and the more specific prefixes generated from it might be 64 bits long (“/64”). In the following example, the leftmost 48 bits of all the specific prefixes will be the same, and they are the same as the general prefix itself. The next 16 bits are all different.

```
 General prefix: 2001:DB8:2222::/48
Specific prefix: 2001:DB8:2222:0000::/64
Specific prefix: 2001:DB8:2222:0001::/64
Specific prefix: 2001:DB8:2222:4321::/64
Specific prefix: 2001:DB8:2222:7744::/64
```

General prefixes can be defined in several ways:

- Manually
- Based on a 6to4 interface
- Dynamically, from a prefix received by a Dynamic Host Configuration Protocol (DHCP) for IPv6 prefix delegation client

More specific prefixes, based on a general prefix, can be used when configuring IPv6 on an interface.

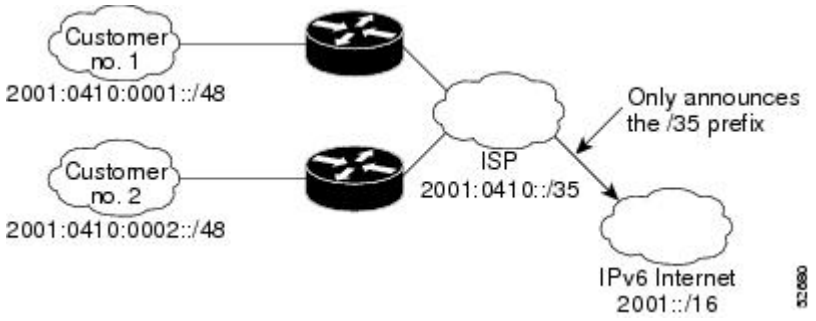
DHCP for IPv6 Prefix Delegation

DHCP for IPv6 can be used in environments to deliver stateful and stateless information. For further information about this feature, see *Implementing DHCP for IPv6*.

IPv6 Prefix Aggregation

The aggregatable nature of the IPv6 address space enables an IPv6 addressing hierarchy. For example, an enterprise can subdivide a single IPv6 prefix from a service provider into multiple, longer prefixes for use within its internal network. Conversely, a service provider can aggregate all of the prefixes of its customers into a single, shorter prefix that the service provider can then advertise over the IPv6 internet (see the figure below).

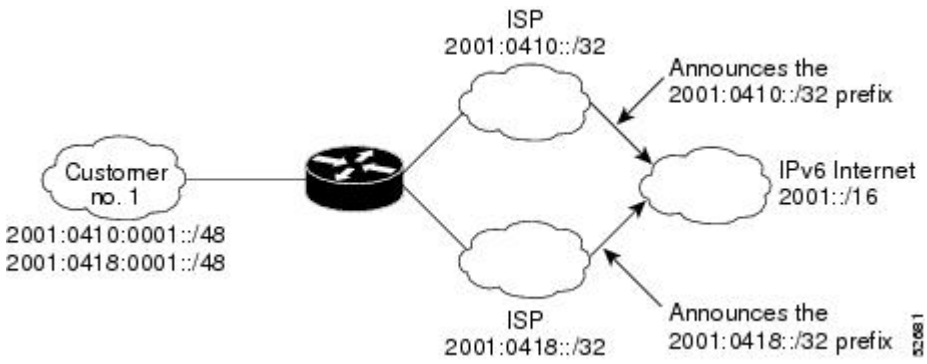
Figure 16 IPv6 Prefix Aggregation



IPv6 Site Multihoming

Multiple IPv6 prefixes can be assigned to networks and hosts. Having multiple prefixes assigned to a network allows that network to connect easily to multiple ISPs without breaking the global routing table (see the figure below).

Figure 17 IPv6 Site Multihoming



IPv6 Data Links

In IPv6 networks, a data link is a network sharing a particular link-local prefix. Data links are networks arbitrarily segmented by a network administrator in order to provide a multilevel, hierarchical routing structure while shielding the subnetwork from the addressing complexity of attached networks. The function of a subnetwork in IPv6 is similar to a subnetwork in IPv4. A subnetwork prefix is associated with one data link; multiple subnetwork prefixes may be assigned to the same data link.

The following data links are supported for IPv6: ATM permanent virtual circuit (PVC) and ATM LANE, dynamic packet transport (DPT), Ethernet, Fast Ethernet, FDDI, Frame Relay PVC, Gigabit Ethernet, Cisco High-Level Data Link Control (HDLC), ISDN, PPP over Packet over SONET (PoS), and serial interfaces.

IPv6 for Cisco Software Support for Wide-Area Networking Technologies

IPv6 for Cisco software supports wide-area networking technologies such as ATM PVCs, Frame Relay PVCs, Cisco HDLC, ISDN, PoS, and serial (synchronous and asynchronous) interface types. These technologies function the same in IPv6 as they do in IPv4.

IPv6 Addresses and PVCs

Broadcast and multicast are used in LANs to map protocol (network layer) addresses to the hardware addresses of remote nodes (hosts and devices). Because using broadcast and multicast to map network layer addresses to hardware addresses in circuit-based WANs such as ATM and Frame Relay networks is difficult to implement, these networks use implicit, explicit, and dynamic mappings for the network layer addresses of remote nodes and the PVCs used to reach the addresses.

Assigning an IPv6 address to an interface by using the **ipv6 address** command defines the IPv6 addresses for the interface and the network that is directly connected to the interface. If only one PVC is terminated on the interface (the interface is a point-to-point interface), there is an implicit mapping between all of the IPv6 addresses on the network and the PVC used to reach the addresses (no additional address mappings are needed). If several PVCs are terminated on the interface (the interface is a point-to-multipoint interface), the **protocol ipv6** command (for ATM networks) or the **frame-relay map ipv6** command (for Frame Relay networks) is used to configure explicit mappings between the IPv6 addresses of the remote nodes and the PVCs used to reach the addresses.

Note Given that IPv6 supports multiple address types, and depending on which applications or protocols are configured on a point-to-multipoint interface, you may need to configure multiple explicit mappings between the IPv6 addresses of the interface and the PVC used to reach the addresses. For example, explicitly mapping both the link-local and global IPv6 address of a point-to-multipoint interface to the PVC on which the interface terminates ensures that the Interior Gateway Protocol (IGP) configured on the interface forwards traffic to and from the PVC correctly.

Routed Bridge Encapsulation for IPv6

Routed bridge encapsulation (RBE) provides a mechanism for routing a protocol from a bridged interface to another routed or bridged interface. RBE for IPv6 can be used on ATM point-to-point subinterfaces that are configured for IPv6 half-bridging. Routing of IP packets and IPv6 half-bridging, bridging, PPP over Ethernet (PPPoE), or other Ethernet 802.3-encapsulated protocols can be configured on the same subinterface.

IPv6 Redirect Messages

The IPv6 Redirect Messages feature enables a device to send ICMP IPv6 neighbor redirect messages to inform hosts of better first hop nodes (devices or hosts) on the path to a destination.

IPv6 on BVI Interfaces for Bridging and Routing

Integrated routing and bridging (IRB) enables users to route a given protocol between routed interfaces and bridge groups or route a given protocol between bridge groups. Specifically, local or unroutable traffic will be bridged among the bridged interfaces in the same bridge group, while routable traffic will be routed to other routed interfaces or bridge groups. If you want both bridging and routing capabilities, IRB is required. If you want only bridging, you must disable routing. To disable the routing function for IPv6, you must configure the **no ipv6 unicast-routing** command.

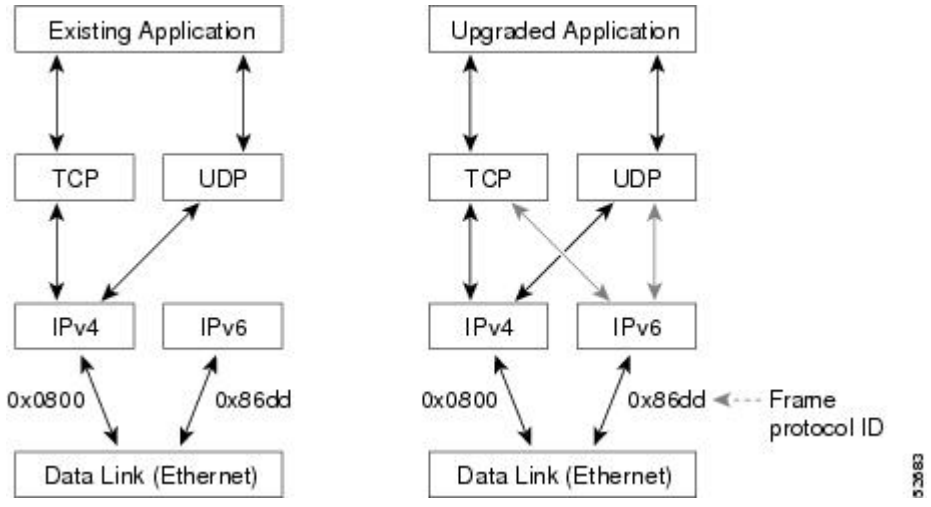
IPv6 is supported in the bridge virtual interface (BVI), which is the IPv4 interface for bridged interfaces. Because bridging is in the data link layer and routing is in the network layer, they have different protocol configuration models to follow. In the basic IPv4 model, for example, all bridged interfaces should belong to the same network, while each routed interface represents a distinct network. Routed traffic is destined for the device, while bridged traffic is never destined for the device. Using BVI avoids the confusion of which protocol configuration model to use when both bridging and routing a given protocol in the same bridge group.

Note BVIs in IPv6 are not supported with Network Address Translation--Protocol Translation (NAT-PT) and wireless interfaces Dot11Radio.

Dual IPv4 and IPv6 Protocol Stacks

The dual IPv4 and IPv6 protocol stack technique can be used to transition to IPv6. It enables gradual, one-by-one upgrades to applications running on nodes. Applications running on nodes are upgraded to make use of the IPv6 protocol stack. Applications that are not upgraded (for example, they support only the IPv4 protocol stack) can coexist with upgraded applications on a node. New and upgraded applications make use of both the IPv4 and IPv6 protocol stacks (see the figure below).

Figure 18 Dual IPv4 and IPv6 Protocol Stack Technique

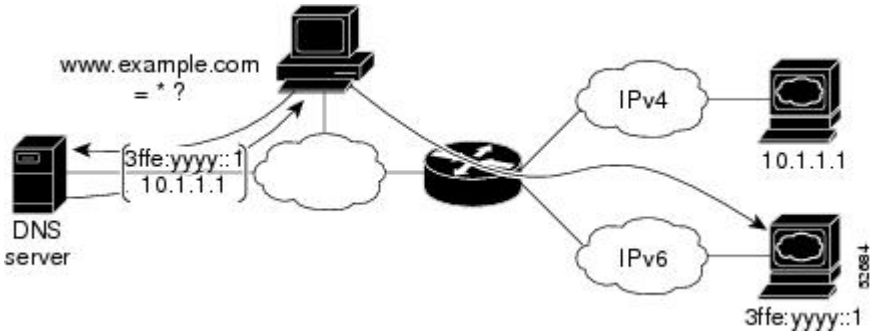


One application program interface (API) supports both IPv4 and IPv6 addresses and DNS requests. An application can be upgraded to the new API and still use only the IPv4 protocol stack. The Cisco software

supports the dual IPv4 and IPv6 protocol stack technique. When an interface is configured with both an IPv4 and an IPv6 address, the interface will forward both IPv4 and IPv6 traffic.

In the figure below, an application that supports dual IPv4 and IPv6 protocol stacks requests all available addresses for the destination hostname www.example.com from a DNS server. The DNS server replies with all available addresses (both IPv4 and IPv6 addresses) for www.example.com. The application chooses an address (in most cases, IPv6 addresses are the default choice), and connects the source node to the destination using the IPv6 protocol stack.

Figure 19 Dual IPv4 and IPv6 Protocol Stack Applications



How to Implement IPv6 Addressing and Basic Connectivity

Configuring IPv6 Addressing and Enabling IPv6 Routing

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type number*
4. Do one of the following:
   - **ipv6 address** *ipv6-prefix/prefix-length* **eui-64**
   - **ipv6 address** *ipv6-prefix/prefix-length* **link-local**
   - **ipv6 address** *ipv6-prefix/prefix-length* **anycast**
   - **ipv6 enable**
5. **exit**
6. **ipv6 unicast-routing**

DETAILED STEPS

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

	Command or Action	Purpose

Step 4	Do one of the following:
	• **ipv6 address** *ipv6-prefix/prefix-length* **eui-64**
	• **ipv6 address** *ipv6-prefix/prefix-length* **link-local**
	• **ipv6 address** *ipv6-prefix/prefix-length* **anycast**
	• **ipv6 enable**
	Example:
	`Device(config-if)# ipv6 address 2001:DB8:0:1::/64 eui-64`
	Example:
	`Device(config-if)# ipv6 address FE80::260:3EFF:FE11:6770 link-local`
	Example:
	`Device(config-if) ipv6 address 2001:DB8:1:1:FFFF:FFFF:FFFF:FFFE/64 anycast`
	Example:
	`Device(config-if)# ipv6 enable`
		Specifies an IPv6 network assigned to the interface and enables IPv6 processing on the interface.
		or
		Specifies an IPv6 address 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.
		or
		Enable IPv6 processing on an interface that has not been configured with an explicit IPv6 address.
		• Specifying the **ipv6 address eui-64** command configures 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.
		• Specifying the **ipv6 address anycast** command adds an IPv6 anycast address.

Step 5	**exit** **Example:** `Device(config-if)# exit`	Exits interface configuration mode, and returns the device to global configuration mode.
Step 6	**ipv6 unicast-routing** **Example:** `Device(config)# ipv6 unicast-routing`	Enables the forwarding of IPv6 unicast datagrams.

- Configuring a Neighbor Discovery Cache Limit, page 31
- Customizing the Parameters for IPv6 Neighbor Discovery, page 33

Configuring a Neighbor Discovery Cache Limit

Configuring a Neighbor Discovery Cache Limit on a Specified Interface

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type number*
4. **ipv6 nd cache interface-limit** *size* [**log** *rate*]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *type number* **Example:** `Device(config)# interface GigabitEthernet 1/0/0`	Specifies an interface type and number, and places the device in interface configuration mode.
Step 4	**ipv6 nd cache interface-limit** *size* [**log** *rate*] **Example:** `Device(config-if)# ipv6 nd cache interface-limit 1`	Configures a Neighbor Discovery cache limit on a specified interface on the device. • Issuing this command overrides any configuration that may have been created by issuing the **ipv6 nd cache interface-limit** in global configuration mode.

Configuring a Neighbor Discovery Cache Limit on All Device Interfaces

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ipv6 nd cache interface-limit** *size* [**log** *rate*]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**ipv6 nd cache interface-limit** *size* [**log** *rate*] **Example:** `Device(config)# ipv6 nd cache interface-limit 4`	Configures a neighbor discovery cache limit on all interfaces on the device.

Customizing the Parameters for IPv6 Neighbor Discovery

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type number*
4. **ipv6 nd nud retry** *base interval max-attempts*
5. **ipv6 nd cache expire** *expire-time-in-seconds* [**refresh**]
6. **ipv6 nd na glean**

DETAILED STEPS

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

	Command or Action	Purpose
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *type number* **Example:** `Device(config)# interface Ethernet 1/0`	Specifies an interface type and number, and places the device in interface configuration mode.
Step 4	**ipv6 nd nud retry** *base interval max-attempts* **Example:** `Device(config-if)# ipv6 nd nud retry 1 1000 3`	Configures the number of times neighbor unreachability detection (NUD) resends neighbor solicitations.
Step 5	**ipv6 nd cache expire** *expire-time-in-seconds* [**refresh**] **Example:** `Device(config-if)# ipv6 nd cache expire 7200`	Configures the length of time before an IPv6 neighbor discovery cache entry expires.
Step 6	**ipv6 nd na glean** **Example:** `Device(config-if)# ipv6 nd na glean`	Configures ND to glean an entry from an unsolicited neighbor advertisement (NA).

Defining and Using IPv6 General Prefixes

General prefixes can be defined in several ways:

- Manually
- Based on a 6to4 interface
- Dynamically, from a prefix received by a DHCP for IPv6 prefix delegation client

More specific prefixes, based on a general prefix, can be used when configuring IPv6 on an interface.

- Defining a General Prefix Manually, page 35
- Defining a General Prefix Based on a 6to4 Interface, page 35
- Defining a General Prefix with the DHCP for IPv6 Prefix Delegation Client Function, page 36
- Using a General Prefix in IPv6, page 36

Defining a General Prefix Manually

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ipv6 general-prefix** *prefix-name* {*ipv6-prefix/prefix-length* | **6to4** *interface-type interface-number*}

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**ipv6 general-prefix** *prefix-name* {*ipv6-prefix/prefix-length*	**6to4** *interface-type interface-number*} **Example:** `Device(config)# ipv6 general-prefix my-prefix 2001:DB8:2222::/48`	Defines a general prefix for an IPv6 address.

Defining a General Prefix Based on a 6to4 Interface

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ipv6 general-prefix** *prefix-name* {*ipv6-prefix/prefix-length* | **6to4** *interface-type interface-number*}

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**ipv6 general-prefix** *prefix-name* {*ipv6-prefix/prefix-length* | **6to4** *interface-type interface-number*} **Example:** `Device(config)# ipv6 general-prefix my-prefix 6to4 ethernet 0`	Defines a general prefix for a 6to4 address.

Defining a General Prefix with the DHCP for IPv6 Prefix Delegation Client Function

You can define a general prefix dynamically using the DHCP for IPv6 prefix delegation client function. For information on how to perform this task, see the Implementing DHCP for IPv6 module.

Using a General Prefix in IPv6

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type number*
4. **ipv6 address** { *ipv6-address/prefix-length* | *prefix-name sub-bits/prefix-length*}

DETAILED STEPS

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

	Command or Action	Purpose
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *type number* **Example:** `Device(config)# interface ethernet 0/0`	Specifies an interface type and number, and places the device in interface configuration mode.
Step 4	**ipv6 address** { *ipv6-address/prefix-length* | *prefix-name sub-bits/ prefix-length*} **Example:** `Device(config-if) ipv6 address my-prefix 2001:DB8:0:7272::/64`	Configures an IPv6 prefix name for an IPv6 address and enables IPv6 processing on the interface.

Configuring an Interface to Support 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; that is, the interface can send and receive data on both IPv4 and IPv6 networks.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ipv6 unicast-routing**
4. **interface** *type number*
5. **ip address** *ip-address mask* [**secondary** [**vrf** *vrf-name*]]
6. **ipv6 address** {*ipv6-address /prefix-length* | *prefix-name sub-bits* / *prefix-length*}

DETAILED STEPS

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

	Command or Action	Purpose
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**ipv6 unicast-routing** **Example:** `Device(config)# ipv6 unicast-routing`	Enables the forwarding of IPv6 unicast datagrams.
Step 4	**interface** *type number* **Example:** `Device(config)# interface ethernet 0`	Specifies the interface type and number, and enters interface configuration mode.
Step 5	**ip address** *ip-address mask* [**secondary** [**vrf** *vrf-name*]] **Example:** `Device(config-if)# ip address 192.168.99.1 255.255.255.0`	Specifies a primary or secondary IPv4 address for an interface.
Step 6	**ipv6 address** {*ipv6-address /prefix-length* | *prefix-name sub-bits / prefix-length*} **Example:** `Device(config-if)# ipv6 address 2001:DB8:c18:1::3/64`	Specifies the IPv6 network assigned to the interface and enables IPv6 processing on the interface.

Customizing IPv6 ICMP Rate Limiting

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ipv6 icmp error-interval** *milliseconds* [*bucketsize*]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**ipv6 icmp error-interval** *milliseconds* [*bucketsize*] **Example:** `Device(config)# ipv6 icmp error-interval 50 20`	Customizes the interval and bucket size for IPv6 ICMP error messages.

Configuring the DRP Extension for Traffic Engineering

Perform this task to configure the DRP extension to RAs, which signals the preference value of a default device.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type number*
4. **ipv6 nd router-preference** {**high** | **medium** | **low**}

DETAILED STEPS

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

	Command or Action	Purpose
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *type number* **Example:** `Device(config)# interface ethernet 0`	Specifies the interface type and number, and enters interface configuration mode.
Step 4	**ipv6 nd router-preference** {**high** | **medium** | **low**} **Example:** `Device(config-if)# ipv6 nd router-preference high`	Configures a DRP for a device on a specific interface.

Configuring Cisco Express Forwarding and Distributed Cisco Express Forwarding Switching for IPv6

Configuring Cisco Express Forwarding Switching on Distributed and Nondistributed Architecture Platforms

Cisco Express Forwarding is designed for nondistributed architecture platforms. Distributed Cisco Express Forwarding is designed for distributed architecture platforms. Nondistributed platforms do not support distributed Cisco Express Forwarding; however, some distributed platforms support both Cisco Express Forwarding and distributed Cisco Express Forwarding.

To enable the device to forward Cisco Express Forwarding and distributed Cisco Express Forwarding traffic, use the **ipv6 unicast-routing** command to configure the forwarding of IPv6 unicast datagrams globally on the device, and use the **ipv6 address** command to configure IPv6 address and IPv6 processing on an interface.

You must enable Cisco Express Forwarding for IPv4 globally on the device by using the **ip cef** command before enabling Cisco Express Forwarding for IPv6 globally on the device.

You must enable distributed Cisco Express Forwarding for IPv4 by using the **ip cef distributed** command before enabling distributed Cisco Express Forwarding for IPv6.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. Do one of the following:
   - **ipv6 cef**
   - **ipv6 cef distributed**
4. **ipv6 cef accounting** [**non-recursive** | **per-prefix** | **prefix-length**]

DETAILED STEPS

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

Step 3	Do one of the following:
	• **ipv6 cef**
	• **ipv6 cef distributed**
	
	Example:
	`Device(config)# ipv6 cef`
	
	Example:
	`Device(config)# ipv6 cef distributed`
		Enables Cisco Express Forwarding globally on the device.
		or
		Enables distributed Cisco Express Forwarding globally on the device.

Command or Action	Purpose
Step 4 **ipv6 cef accounting** [**non-recursive** | **per-prefix** | **prefix-length**] **Example:** `Device(config)# ipv6 cef accounting`	Enables Cisco Express Forwarding and distributed Cisco Express Forwarding network accounting globally on the device. • Network accounting for Cisco Express Forwarding and distributed Cisco Express Forwarding enables you to better understand Cisco Express Forwarding traffic patterns within your network by collecting statistics specific to Cisco Express Forwarding and distributed Cisco Express Forwarding traffic. For example, network accounting for Cisco Express Forwarding and distributed Cisco Express Forwarding enables you to collect information such as the number of packets and bytes switched to a destination or the number of packets switched through a destination. • The optional **per-prefix** keyword enables the collection of the number of packets and bytes express forwarded to an IPv6 destination or IPv6 prefix. • The optional **prefix-length** keyword enables the collection of the number of packets and bytes express forwarded to an IPv6 prefix length. **Note** When Cisco Express Forwarding is enabled globally on the device, accounting information is collected at the RP; when distributed Cisco Express Forwarding is enabled globally on the device, accounting information is collected at the line cards.

Enabling Unicast RPF

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type number*
4. **ipv6 verify unicast source reachable-via** {**rx** | **any**} [**allow-default**] [**allow-self-ping**] [*access-list-name*]

DETAILED STEPS

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

	Command or Action	Purpose
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *type number* **Example:** `Device(config)# interface atm 0`	Specifies an interface type and number, and places the device in interface configuration mode.
Step 4	**ipv6 verify unicast source reachable-via** {**rx** | **any**} [**allow-default**] [**allow-self-ping**] [*access-list-name*] **Example:** `Device(config-if)# ipv6 verify unicast source reachable-via any`	Verifies that a source address exists in the FIB table and enables uRPF.

Mapping Hostnames to IPv6 Addresses

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ipv6 host** *name* [*port*] *ipv6-address1* [*ipv6-address2...ipv6-address4*]
4. Do one of the following:
   - **ip domain name** [**vrf** *vrf-name*] *name*
   - **ip domain list** [**vrf** *vrf-name*] *name*
5. **ip name-server** [**vrf** *vrf-name*] *server-address1* [*server-address2...server-address6*]
6. **ip domain-lookup**

DETAILED STEPS

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

Command or Action	Purpose
Step 2 **configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3 **ipv6 host** *name* [*port*] *ipv6-address1* [*ipv6-address2...ipv6-address4*] **Example:** `Device(config)# ipv6 host cisco-sj 2001:DB8:20:1::12`	Defines a static hostname-to-address mapping in the hostname cache. • You may find it easier to refer to network devices by symbolic names rather than numerical addresses (services such as Telnet can use hostnames or addresses). Hostnames and IPv6 addresses can be associated with one another through static or dynamic means. • Manually assigning hostnames to addresses is useful when dynamic mapping is not available.

Step 4 Do one of the following:
• **ip domain name** [**vrf** *vrf-name*] *name*
• **ip domain list** [**vrf** *vrf-name*] *name*
 	
Example:
`Device(config)# ip domain-name cisco.com`
 	
Example:
`Device(config)# ip domain list cisco1.com`
	(Optional) Defines a default domain name that the Cisco software will use to complete unqualified hostnames.
	or
	(Optional) Defines a list of default domain names to complete unqualified hostnames.
	• You can specify a default domain name that the Cisco software will use to complete domain name requests. You can specify either a single domain name or a list of domain names. Any hostname that does not contain a complete domain name will have the default domain name you specify appended to it before the name is looked up.
	Note The **ip domain name** and **ip domain list** commands are used to specify default domain names that can be used by both IPv4 and IPv6.

Step 5 **ip name-server** [**vrf** *vrf-name*] *server-address1* [*server-address2...server-address6*] **Example:** `Device(config)# ip name-server 2001:DB8::250:8bff:fee8:f800 2001:DB8:0:f004::1`	Specifies one or more hosts that supply name information. • Specifies one or more hosts (up to six) that can function as a name server to supply name information for DNS. **Note** The *server-address* argument can be either an IPv4 or IPv6 address.
Step 6 **ip domain-lookup** **Example:** `Device(config)# ip domain-lookup`	Enables DNS-based address translation. • DNS is enabled by default.

Mapping IPv6 Addresses to IPv6 ATM and Frame Relay Interfaces

Perform this task to map IPv6 addresses to ATM and Frame Relay PVCs. Specifically, the steps in this section explain how to explicitly map IPv6 addresses to the ATM and Frame Relay PVCs used to reach the addresses.

Note This task shows how to configure both ATM and Frame Relay PVCs. Many of the steps are labeled optional because many networks will require only one type of PVC to be configured. The steps in this section are not applicable to ATM LANE.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type number*
4. **pvc** [*name*] *vpi/vci* [**ces** | **ilmi** | **qsaal** | **smds** | **l2transport**]
5. **protocol ipv6** *ipv6-address* [[**no**] **broadcast**]
6. **exit**
7. **ipv6 address** *ipv6-address/prefix-length* **link-local**
8. **exit**
9. **interface** *type number*
10. **frame-relay map ipv6** *ipv6-address dlci* [**broadcast**] [**cisco**] [**ietf**] [**payload-compression** {**packet-by-packet** | **frf9 stac** [*hardware-options*] | **data-stream stac** [*hardware-options*]}]
11. **ipv6 address** *ipv6-address/prefix-length* **link-local**

DETAILED STEPS

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

	Command or Action	Purpose
Step 3	**interface** *type number* **Example:** `Router(config)# interface atm 0`	Specifies an interface type and number, and places the router in interface configuration mode.
Step 4	**pvc** [*name*] *vpi/vci* [**ces** | **ilmi** | **qsaal** | **smds** | **l2transport**] **Example:** `Router(config-if)# pvc 1/32`	(Optional) Creates or assigns a name to an ATM PVC and places the router in ATM VC configuration mode.
Step 5	**protocol ipv6** *ipv6-address* [[**no**] **broadcast**] **Example:** `Router(config-if-atm-vc)# protocol ipv6 2001:DB8:2222:1003::45`	(Optional) Maps the IPv6 address of a remote node to the PVC used to reach the address. • The *ipv6-address* argument must be in the form documented in RFC 2373 where the address is specified in hexadecimal using 16-bit values between colons. • The optional [**no**] **broadcast** keywords indicate whether the map entry should be used when IPv6 multicast packets (not broadcast packets) are sent to the interface. Pseudobroadcasting is supported. The [**no**] **broadcast** keywords in the **protocol ipv6** command take precedence over the **broadcast** command configured on the same ATM PVC.
Step 6	**exit** **Example:** `Router(config-if-atm-vc)# exit`	Exits ATM VC configuration mode, and returns the router to interface configuration mode.
Step 7	**ipv6 address** *ipv6-address/prefix-length* **link-local** **Example:** `Router(config-if)# ipv6 address 2001:DB8:2222:1003::72/64 link-local`	Specifies an IPv6 network assigned to the interface and enables IPv6 processing on the interface. • In the context of this task, a link-local address of the node at the other end of the link is required for the IGP used in the network. • 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.
Step 8	**exit** **Example:** `Router(config-if)# exit`	Exits interface configuration mode, and returns the router to global configuration mode.

	Command or Action	Purpose
Step 9	**interface** *type number* **Example:** `Router(config)# interface serial 3`	Specifies an interface type and number, and places the router in interface configuration mode.
Step 10	**frame-relay map ipv6** *ipv6-address dlci* [**broadcast**] [**cisco**] [**ietf**] [**payload-compression** {**packet-by-packet** | **frf9 stac** [*hardware-options*] | **data-stream stac** [*hardware-options*]}] **Example:** `Router(config-if)# frame-relay map ipv6 FE80::E0:F727:E400:A 17 broadcast`	(Optional) Maps the IPv6 address of a remote node to the data-link connection identifier (DLCI) of the PVC used to reach the address.
Step 11	**ipv6 address** *ipv6-address/prefix-length* **link-local** **Example:** `Router(config-if)# ipv6 address 2001:DB8:2222:1044::46/64 link-local`	Specifies an IPv6 network assigned to the interface and enables IPv6 processing on the interface. • In the context of this task, a link-local address of the node at the other end of the link is required for the IGP used in the network. • 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.

Displaying IPv6 Redirect Messages

SUMMARY STEPS

1. **enable**
2. **show ipv6 interface** [**brief**] [*type number*] [**prefix**]
3. **show ipv6 neighbors** [*interface-type interface-number* | *ipv6-address* | *ipv6-hostname* | **statistics**]
4. **show ipv6 route** [*ipv6-address* | *ipv6-prefix* / *prefix-length* | *protocol* | *interface-type interface-number*]
5. **show ipv6 traffic**
6. **show atm map**
7. **show hosts** [**vrf** *vrf-name* | **all** | *hostname* | **summary**]
8. **show running-config**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device# enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**show ipv6 interface** [**brief**] [*type number*] [**prefix**] **Example:** `Device# show ipv6 interface ethernet 0`	Displays the usability status of interfaces configured for IPv6.
Step 3	**show ipv6 neighbors** [*interface-type interface-number* | *ipv6-address* | *ipv6-hostname* | **statistics**] **Example:** `Device# show ipv6 neighbors ethernet 2`	Displays IPv6 neighbor discovery cache information.
Step 4	**show ipv6 route** [*ipv6-address* | *ipv6-prefix* / *prefix-length* | *protocol* | *interface-type interface-number*] **Example:** `Device# show ipv6 route`	Displays the current contents of the IPv6 routing table.
Step 5	**show ipv6 traffic** **Example:** `Device# show ipv6 traffic`	Displays statistics about IPv6 traffic.
Step 6	**show atm map** **Example:** `Device# show atm map`	Displays the list of all configured ATM static maps to remote hosts on an ATM network and on ATM bundle maps.
Step 7	**show hosts** [**vrf** *vrf-name* | **all** | *hostname* | **summary**] **Example:** `Device# show hosts`	Displays the default domain name, the style of name lookup service, a list of name server hosts, and the cached list of hostnames and addresses.

Command or Action	Purpose
Step 8 **show running-config** **Example:** `Device# show running-config`	Displays the current configuration running on the device.

- Examples, page 49

Examples

Sample Output from the show ipv6 interface Command

In the following example, the **show ipv6 interface** command is used to verify that IPv6 addresses are configured correctly for Ethernet interface 0. Information is also displayed about the status of IPv6 neighbor redirect messages, IPv6 neighbor discovery messages, and stateless autoconfiguration.

```
Router# show ipv6 interface ethernet 0

Ethernet0 is up, line protocol is up
  IPv6 is stalled, link-local address is FE80::1
  Global unicast address(es):
    2001:DB8:2000::1, subnet is 2001:DB8:2000::/64
    2001:DB8:3000::1, subnet is 2001:DB8:3000::/64
  Joined group address(es):
    FF02::1
    FF02::2
    FF02::1:FF00:1
  MTU is 1500 bytes
  ICMP error messages limited to one every 100 milliseconds
  ICMP redirects are enabled
  ND DAD is enabled, number of DAD attempts: 1
  ND reachable time is 30000 milliseconds
  ND advertised reachable time is 0 milliseconds
  ND advertised retransmit interval is 0 milliseconds
  ND router advertisements are sent every 200 seconds
  ND router advertisements live for 1800 seconds
  Hosts use stateless autoconfig for addresses.
```

Sample Output from the show ipv6 neighbors Command

In the following example, the **show ipv6 neighbors** command is used to display IPv6 neighbor discovery cache information. A hyphen (-) in the Age field of the command output indicates a static entry. The following example displays IPv6 neighbor discovery cache information for Ethernet interface 2:

```
Router# show ipv6 neighbors ethernet 2

IPv6 Address                              Age Link-layer Addr State Interface
2001:DB8:0:4::2                            0 0003.a0d6.141e  REACH Ethernet2
FE80::XXXX:A0FF:FED6:141E                  0 0003.a0d6.141e  REACH Ethernet2
2001:DB8:1::45a                            - 0002.7d1a.9472  REACH Ethernet2
```

Sample Output from the show ipv6 route Command

When the *ipv6-address* or *ipv6-prefix*/*prefix-length* argument is specified, only route information for that address or network is displayed. The following is sample output from the **show ipv6 route** command when entered with the IPv6 prefix 2001:DB8::/35:

```
Router# show ipv6 route 2001:DB8::/35

IPv6 Routing Table - 261 entries
Codes: C - Connected, L - Local, S - Static, R - RIP, B - BGP
I1 - ISIS L1, I2 - ISIS L2, IA - ISIS interarea
B 2001:DB8::/35 [20/3]
  via FE80::60:5C59:9E00:16, Tunnel1
```

Sample Output from the show ipv6 traffic Command

In the following example, the **show ipv6 traffic** command is used to display ICMP rate-limited counters:

```
Router# show ipv6 traffic

ICMP statistics:
  Rcvd: 188 input, 0 checksum errors, 0 too short
        0 unknown info type, 0 unknown error type
        unreach: 0 routing, 0 admin, 0 neighbor, 0 address, 0 port
        parameter: 0 error, 0 header, 0 option
        0 hopcount expired, 0 reassembly timeout,0 too big
        0 echo request, 0 echo reply
        0 group query, 0 group report, 0 group reduce
        1 router solicit, 175 router advert, 0 redirects
        0 neighbor solicit, 12 neighbor advert
  Sent: 7376 output, 56 rate-limited
        unreach: 0 routing, 15 admin, 0 neighbor, 0 address, 0 port
        parameter: 0 error, 0 header, 0 option
        0 hopcount expired, 0 reassembly timeout,0 too big
        15 echo request, 0 echo reply
        0 group query, 0 group report, 0 group reduce
        0 router solicit, 7326 router advert, 0 redirects
        2 neighbor solicit, 22 neighbor advert
```

Sample Output from the show frame-relay map Command

In the following example, the **show frame-relay map** command is used to verify that the IPv6 address of a remote node is mapped to the DLCI of the PVC used to reach the address. The following example shows that the link-local and global IPv6 addresses (FE80::E0:F727:E400:A and 2001:DB8:2222:1044::73; FE80::60:3E47:AC8:8 and 2001.DB8:2222:1044::72) of two remote nodes are explicitly mapped to DLCI 17 and DLCI 19, respectively. Both DLCI 17 and DLCI 19 are terminated on interface serial 3 of this node; therefore, interface serial 3 of this node is a point-to-multipoint interface.

```
Router# show frame-relay map

Serial3 (up): ipv6 FE80::E0:F727:E400:A dlci 17(0x11,0x410), static,
              broadcast, CISCO, status defined, active
Serial3 (up): ipv6 2001:DB8:2222:1044::72 dlci 19(0x13,0x430), static,
              CISCO, status defined, active
Serial3 (up): ipv6 2001:DB8:2222:1044::73 dlci 17(0x11,0x410), static,
              CISCO, status defined, active
Serial3 (up): ipv6 FE80::60:3E47:AC8:8 dlci 19(0x13,0x430), static,
              broadcast, CISCO, status defined, active
```

Sample Output from the show atm map Command

In the following example, the **show atm map** command is used to verify that the IPv6 address of a remote node is mapped to the PVC used to reach the address. The following example shows that the link-local and

global IPv6 addresses (FE80::60:3E47:AC8:C and 2001:DB8:2222:1003::72, respectively) of a remote node are explicitly mapped to PVC 1/32 of ATM interface 0:

```
Router# show atm map

Map list ATM0pvc1 : PERMANENT
ipv6 FE80::60:3E47:AC8:C maps to VC 1, VPI 1, VCI 32, ATM0
        , broadcast
ipv6 2001:DB8:2222:1003::72 maps to VC 1, VPI 1, VCI 32, ATM0
```

Sample Output from the show hosts Command

The state of the name lookup system on the DHCP for IPv6 client can be displayed with the **show hosts** command:

```
Router# show hosts

Default domain is not set
Domain list:example.com
Name/address lookup uses domain service
Name servers are 2001:DB8:A:B::1, 2001:DB8:3000:3000::42
Codes:UN - unknown, EX - expired, OK - OK, ?? - revalidate
       temp - temporary, perm - permanent
       NA - Not Applicable None - Not defined
Host                    Port  Flags      Age Type   Address(es)
sdfasfd                 None  (temp, UN)  0  IPv6
```

Sample Output from the show running-config Command

In the following example, the **show running-config** command is used to verify that IPv6 processing of packets is enabled globally on the router and on applicable interfaces, and that an IPv6 address is configured on applicable interfaces:

```
Router# show running-config

Building configuration...
Current configuration : 22324 bytes
!
! Last configuration change at 14:59:38 PST Tue Jan 16 2001
! NVRAM config last updated at 04:25:39 PST Tue Jan 16 2001 by bird
!
hostname Router
!
ipv6 unicast-routing
!
interface Ethernet0
 no ip route-cache
 no ip mroute-cache
 no keepalive
 media-type 10BaseT
    ipv6 address 2001:DB8:0:1::/64 eui-64
!
```

In the following example, the **show running-config** command is used to verify that Cisco Express Forwarding and network accounting for Cisco Express Forwarding have been enabled globally on a nondistributed architecture platform, and that Cisco Express Forwarding has been enabled on an IPv6 interface. The following output shows that both that Cisco Express Forwarding and network accounting for Cisco Express Forwarding have been enabled globally on the router, and that Cisco Express Forwarding has also been enabled on Ethernet interface 0:

```
Router# show running-config

Building configuration...
Current configuration : 22324 bytes
!
```

```
! Last configuration change at 14:59:38 PST Tue Jan 16 2001
! NVRAM config last updated at 04:25:39 PST Tue Jan 16 2001 by bird
!
hostname Router
!
ip cef
ipv6 unicast-routing
ipv6 cef
ipv6 cef accounting prefix-length
!
!
interface Ethernet0
 ip address 10.4.9.11 255.0.0.0
 media-type 10BaseT
 ipv6 address 2001:DB8:C18:1::/64 eui-64
!
```

In the following example, the **show running-config** command is used to verify that distributed Cisco Express Forwarding and network accounting for distributed Cisco Express Forwarding have been enabled globally on a distributed architecture platform, such as the Cisco 7500 series routers. The following example shows that both distributed Cisco Express Forwarding and network accounting for Cisco Express Forwarding have been enabled globally on the router.

Note Distributed Cisco Express Forwarding is enabled by default on the GSRs and disabled by default on the Cisco 7500 series routers. Therefore, output from the **show running-config** command on the GSRs does not show whether distributed Cisco Express Forwarding is configured globally on the router. The following output is from a Cisco 7500 series router.

```
Router# show running-config

Building configuration...
Current configuration : 22324 bytes
!
! Last configuration change at 14:59:38 PST Tue Jan 16 2001
! NVRAM config last updated at 04:25:39 PST Tue Jan 16 2001 by bird
!
hostname Router
!
ip cef distributed
ipv6 unicast-routing
ipv6 cef distributed
ipv6 cef accounting prefix-length
```

In the following example, the **show running-config** command is used to verify static hostname-to-address mappings, default domain names, and name servers in the hostname cache, and to verify that the DNS service is enabled:

```
Router# show running-config

Building configuration...
!
ipv6 host cisco-sj 2001:DB8:20:1::12
!
ip domain-name cisco.com
ip domain-lookup
ip name-server 2001:DB8:C01F:768::1
```

Configuration Examples for Implementing IPv6 Addressing and Basic Connectivity

Example: IPv6 Addressing and IPv6 Routing Configuration

In this example, IPv6 is enabled on the device with both a link-local address and a global address based on the IPv6 prefix 2001:DB8:c18:1::/64. The EUI-64 interface ID is used in the low-order 64 bits of both addresses. Output from the **show ipv6 interface** command is included to show how the interface ID (260:3EFF:FE47:1530) is appended to the link-local prefix FE80::/64 of Ethernet interface 0.

```
ipv6 unicast-routing
interface ethernet 0
  ipv6 address 2001:DB8:c18:1::/64 eui-64

Device# show ipv6 interface ethernet 0

Ethernet0 is up, line protocol is up
 IPv6 is enabled, link-local address is FE80::260:3EFF:FE47:1530
  Global unicast address(es):
    2001:DB8:C18:1:260:3EFF:FE47:1530, subnet is 2001:DB8:C18:1::/64
  Joined group address(es):
    FF02::1
    FF02::2
    FF02::1:FF47:1530
    FF02::9
  MTU is 1500 bytes
  ICMP error messages limited to one every 500 milliseconds
  ND reachable time is 30000 milliseconds
  ND advertised reachable time is 0 milliseconds
  ND advertised retransmit interval is 0 milliseconds
  ND router advertisements are sent every 200 seconds
  ND router advertisements live for 1800 seconds
  Hosts use stateless autoconfig for addresses.
```

In the following example, multiple IPv6 global addresses within the prefix 2001:DB8::/64 are configured on Ethernet interface 0:

```
interface ethernet 0
 ipv6 address 2001:DB8::1/64
 ipv6 address 2001:DB8::/64 eui-64
```


Example: Customizing the Parameters for IPv6 Neighbor Discovery

In the following example, IPv6 ND NA gleaning is enabled and the IPv6 ND cache expiry is set to 7200 seconds (2 hours):

```
interface Port-channel189
```

```
 no ip address
 ipv6 address FC07::789:1:0:0:3/64
 ipv6 nd reachable-time 2700000
 ipv6 nd na glean
 ipv6 nd cache expire 7200
 no ipv6 redirects
 standby version 2
 standby 2 ipv6 FC07::789:1:0:0:1/64
 standby 2 priority 150
 standby 2 preempt
```

Example: Dual-Protocol Stack Configuration

This example shows how to enable the forwarding of IPv6 unicast datagrams globally on the device and configures Ethernet interface 0 with both an IPv4 address and an IPv6 address:

```
ipv6 unicast-routing
interface Ethernet 0
  ip address 192.168.99.1 255.255.255.0
  ipv6 address 2001:DB8:c18:1::3/64
```

Example: IPv6 ICMP Rate Limiting Configuration

The following example shows an interval of 50 milliseconds and a bucket size of 20 tokens being configured for IPv6 ICMP error messages:

```
ipv6 icmp error-interval 50 20
```

Example: Cisco Express Forwarding and Distributed Cisco Express Forwarding Configuration

In the following example, both Cisco Express Forwarding for IPv6 and network accounting for Cisco Express Forwarding for IPv6 have been enabled globally on a nondistributed architecture device, and Cisco Express Forwarding for IPv6 has been enabled on Ethernet interface 0. The example also shows that the forwarding of IPv6 unicast datagrams has been configured globally on the device with the **ipv6 unicast-routing** command, an IPv6 address has been configured on Ethernet interface 0 with the **ipv6 address** command, and Cisco Express Forwarding for IPv4 has been configured globally on the device with the **ip cef** command.

```
ip cef
ipv6 unicast-routing
ipv6 cef
ipv6 cef accounting prefix-length
interface Ethernet0
 ip address 10.4.9.11 255.0.0.0
 media-type 10BaseT
 ipv6 address 2001:DB8:C18:1::/64 eui-64
```

In the following example, both distributed Cisco Express Forwarding for IPv6 and network accounting for distributed Cisco Express Forwarding for IPv6 have been enabled globally on a distributed architecture device. The forwarding of IPv6 unicast datagrams has been configured globally on the device with the **ipv6 unicast-routing** command and distributed Cisco Express Forwarding for IPv4 has been configured globally on the device with the **ip cef distributed** command.

```
ip cef distributed
ipv6 unicast-routing
ipv6 cef distributed
ipv6 cef accounting prefix-length
```

Example: Hostname-to-Address Mappings Configuration

The following example defines two static hostname-to-address mappings in the hostname cache, establishes a domain list with several alternate domain names to complete unqualified hostnames, specifies host 2001:DB8::250:8bff:fee8:f800 and host 2001:DB8:0:f004::1 as the name servers, and reenables the DNS service:

```
ipv6 host cisco-sj 2001:DB8:700:20:1::12
ipv6 host cisco-hq 2001:DB8:768::1 2001:DB8:20:1::22
ip domain list domain1-list.com
ip domain list serviceprovider2-name.com
ip domain list college2-name.edu
ip name-server 2001:DB8::250:8bff:fee8:f800 2001:DB8:0:f004::1
ip domain-lookup
```

Examples: IPv6 Address to ATM and Frame Relay PVC Mapping Configuration

Example: IPv6 ATM PVC Mapping Configuration (Point-to-Point Interface)

In the following example, two nodes named Router 1 and Router 2 are connected by a single PVC. The point-to-point subinterface ATM0.132 is used on both nodes to terminate the PVC; therefore, the mapping between the IPv6 addresses of both nodes and the PVC is implicit (no additional mappings are required).

Router 1 Configuration

```
interface ATM 0
 no ip address
!
interface ATM 0.132 point-to-point
 pvc 1/32
 encapsulation aal5snap
 !
 ipv6 address 2001:DB8:2222:1003::72/64
```

Router 2 Configuration

```
interface ATM 0
 no ip address
!
interface ATM 0.132 point-to-point
 pvc 1/32
 encapsulation aal5snap
 !
 ipv6 address 2001:DB8:2222:1003::45/64
```

Example: IPv6 ATM PVC Mapping Configuration (Point-to-Multipoint Interface)

In the following example, the same two nodes (Router 1 and Router 2) from the previous example are connected by the same PVC. In this example, however, the point-to-multipoint interface ATM0 is used on

both nodes to terminate the PVC; therefore, explicit mappings are required between the link-local and global IPv6 addresses of interface ATM0 on both nodes and the PVC. Additionally, ATM pseudobroadcasts are enabled on the link-local address of interface ATM0 on both nodes. The link-local address specified here is the link-local address of the other end of the PVC.

Router 1 Configuration

```
interface ATM 0
 no ip address
 pvc 1/32
 protocol ipv6 2001:DB8:2222:1003::45
 protocol ipv6 FE80::60:2FA4:8291:2 broadcast
 encapsulation aal5snap
 !
 ipv6 address 2001:DB8:2222:1003::72/64
```

Router 2 Configuration

```
interface ATM 0
 no ip address
 pvc 1/32
 protocol ipv6 FE80::60:3E47:AC8:C broadcast
 protocol ipv6 2001:DB8:2222:1003::72
 encapsulation aal5snap
 !
 ipv6 address 2001:DB8:2222:1003::45/64
```

Example: IPv6 Frame Relay PVC Mapping Configuration (Point-to-Point Interface)

In the following example, three nodes named Router A, Router B, and Router C make up a fully meshed network. Each node is configured with two PVCs, which provide an individual connection to each of the other two nodes. Each PVC is configured on a different point-to-point subinterface, which creates three unique IPv6 networks (2001:DB8:2222:1017:/64, 2001:DB8:2222:1018::/64, and 2001:DB8:2222:1019::/64). Therefore, the mappings between the IPv6 addresses of each node and the DLCI (DLCI 17, 18, and 19) of the PVC used to reach the addresses are implicit (no additional mappings are required).

Note Given that each PVC in the following example is configured on a different point-to-point subinterface, the configuration in the following example can also be used in a network that is not fully meshed. Additionally, configuring each PVC on a different point-to-point subinterface can help simplify your routing protocol configuration. However, the configuration in the following example requires more than one IPv6 network, whereas configuring each PVC on point-to-multipoint interfaces requires only one IPv6 network.

Router A Configuration

```
interface Serial 3
 encapsulation frame-relay
!
interface Serial3.17 point-to-point
 description to Router B
 ipv6 address 2001:DB8:2222:1017::46/64
 frame-relay interface-dlci 17
!
interface Serial 3.19 point-to-point
 description to Router C
 ipv6 address 2001:DB8:2222:1019::46/64
 frame-relay interface-dlci 19
```

Router B Configuration

```
interface Serial 5
 encapsulation frame-relay
!
interface Serial5.17 point-to-point
 description to Router A
 ipv6 address 2001:DB8:2222:1017::73/64
 frame-relay interface-dlci 17
!
interface Serial5.18 point-to-point
 description to Router C
 ipv6 address 2001:DB8:2222:1018::73/64
 frame-relay interface-dlci 18
```

Router C Configuration

```
interface Serial 0
 encapsulation frame-relay
!
interface Serial0.18 point-to-point
 description to Router B
 ipv6 address 2001:DB8:2222:1018::72/64
 frame-relay interface-dlci 18
!
interface Serial0.19 point-to-point
 description to Router A
 ipv6 address 2001:DB8:2222:1019::72/64
 frame-relay interface-dlci 19
```

Example: IPv6 Frame Relay PVC Mapping Configuration (Point-to-Multipoint Interface)

In the following example, the same three nodes (Router A, Router B, and Router C) from the previous example make up a fully meshed network and each node is configured with two PVCs (which provide an individual connection to each of the other two nodes). However, the two PVCs on each node in the following example are configured on a single interface (serial 3, serial 5, and serial 10, respectively), which makes each interface a point-to-multipoint interface. Therefore, explicit mappings are required between the link-local and global IPv6 addresses of each interface on all three nodes and the DLCI (DLCI 17, 18, and 19) of the PVC used to reach the addresses.

Router A Configuration

```
interface Serial 3
 encapsulation frame-relay
 ipv6 address 2001:DB8:2222:1044::46/64
 frame-relay map ipv6 FE80::E0:F727:E400:A 17 broadcast
 frame-relay map ipv6 FE80::60:3E47:AC8:8 19 broadcast
 frame-relay map ipv6 2001:DB8:2222:1044::72 19
 frame-relay map ipv6 2001:DB8:2222:1044::73 17
```

Router B Configuration

```
interface Serial 5
 encapsulation frame-relay
 ipv6 address 2001:DB8:2222:1044::73/64
 frame-relay map ipv6 FE80::60:3E59:DA78:C 17 broadcast
 frame-relay map ipv6 FE80::60:3E47:AC8:8 18 broadcast
 frame-relay map ipv6 2001:DB8:2222:1044::46 17
 frame-relay map ipv6 2001:DB8:2222:1044::72 18
```

Router C Configuration

```
interface Serial 10
 encapsulation frame-relay
 ipv6 address 2001:DB8:2222:1044::72/64
 frame-relay map ipv6 FE80::60:3E59:DA78:C 19 broadcast
 frame-relay map ipv6 FE80::E0:F727:E400:A 18 broadcast
 frame-relay map ipv6 2001:DB8:2222:1044::46 19
 frame-relay map ipv6 2001:DB8:2222:1044::73 18
```

Additional References

Related Documents

Related Topic	Document Title
IPv6 supported feature list	"Start Here: Cisco IOS Software Release Specifics for IPv6 Features," *Cisco IOS IPv6 Configuration Guide*
IPv6 commands: complete command syntax, command mode, command history, defaults, usage guidelines, and examples	*Cisco IOS IPv6 Command Reference*
IPv6 DHCP description and configuration	"Implementing DHCP for IPv6," *Cisco IOS IPv6 Configuration Guide*
IPv4 addressing configuration tasks	"Configuring IPv4 Addresses," *Cisco IOS IP Addressing Services Configuration Guide*
IPv4 services configuration tasks	"Configuring IP Services," *Cisco IOS IP Application Services Configuration Guide*
IPv4 addressing commands	*Cisco IOS IP Addressing Services Command Reference*
IPv4 IP services commands	*Cisco IOS IP Application Services Command Reference*
Stateful switchover	"Stateful Switchover," *Cisco IOS High Availability Configuration Guide*
Switching configuration tasks	*Cisco IOS IP Switching Configuration Guide*
Switching commands	*Cisco IOS IP Switching Command Reference*

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
None	To locate and download MIBs for selected platforms, Cisco software releases, and feature sets, use Cisco MIB Locator found at the following URL: http://www.cisco.com/go/mibs

RFCs

RFCs	Title
RFC 1981	*Path MTU Discovery for IP version 6*
RFC 2373	*IP Version 6 Addressing Architecture*
RFC 2374	*An Aggregatable Global Unicast Address Format*
RFC 2460	*Internet Protocol, Version 6 (IPv6) Specification*
RFC 2461	*Neighbor Discovery for IP Version 6 (IPv6)*
RFC 2462	*IPv6 Stateless Address Autoconfiguration*
RFC 2463	*Internet Control Message Protocol (ICMPv6) for the Internet Protocol Version 6 (IPv6) Specification*
RFC 2464	*Transmission of IPv6 Packets over Ethernet Networks*
RFC 2467	*Transmission of IPv6 Packets over FDDI Networks*
RFC 2472	*IP Version 6 over PPP*
RFC 2492	*IPv6 over ATM Networks*
RFC 2590	*Transmission of IPv6 Packets over Frame Relay Networks Specification*
RFC 2684	*Multiprotocol Encapsulation over ATM Adaptation Layer 5*
RFC 3152	*Delegation of IP6.ARPA*
RFC 3162	*RADIUS and IPv6*
RFC 3513	*Internet Protocol Version 6 (IPv6) Addressing Architecture*
RFC 3596	*DNS Extensions to Support IP version 6*
RFC 3879	*Deprecating Site Local Addresses*

RFCs	Title
RFC 4193	*Unique Local IPv6 Unicast Addresses*

Technical Assistance

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

Feature Information for Implementing IPv6 Addressing and Basic Connectivity

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Table 5 Feature Information for Implementing IPv6 Addressing and Basic Connectivity

Feature Name	Releases	Feature Information
IPv6—Anycast Address	12.2(25)S 12.2(28)SB 12.2(25)SG 12.2(33)SRA 12.2(33)SXH 12.3(4)T 12.4 12.4(2)T	An anycast address is an address that is assigned to a set of interfaces that typically belong to different nodes.
IPv6—Base Protocols High Availability	12.2(33)SRE	IPv6 neighbor discovery supports SSO.

Feature Name	Releases	Feature Information
IPv6—ICMP Rate Limiting	12.2(8)T 12.3 12.3(2)T 12.4 12.4(2)T	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.
IPv6—ICMPv6	12.0(22)S 12.2(14)S 12.2(28)SB 12.2(25)SG 12.2(33)SRA 12.2(17a)SX1 12.2(2)T 12.3 12.3(2)T 12.4 12.4(2)T	ICMP for IPv6 generates error messages, such as ICMP destination unreachable messages, and informational messages, such as ICMP echo request and reply messages. Additionally, ICMP packets in IPv6 are used in the IPv6 neighbor discovery process, path MTU discovery, and the MLD protocol for IPv6.
IPv6—ICMPv6 Redirect	12.0(22)S 12.2(14)S 12.2(28)SB 12.2(25)SG 12.2(33)SRA 12.2(17a)SX1 12.2(4)T 12.3 12.3(2)T 12.4 12.4(2)T	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.
IPv6—IPv6 Default Router Preferences	12.2(33)SB 12.2(33)SRA 12.4(2)T 12.2(33)SXH 15.0(1)S	The DRP extension provides a coarse preference metric (low, medium, or high) for default routers.

Feature Name	Releases	Feature Information
IPv6—IPv6 MTU Path Discovery	12.0(22)S 12.2(14)S 12.2(28)SB 12.2(25)SG 12.2(33)SRA 12.2(17a)SX1 12.2(2)T 12.3 12.3(2)T 12.4 12.4(2)T 15.0(1)S	Path MTU discovery in IPv6 allows a host to dynamically discover and adjust to differences in the MTU size of every link along a given data path.
IPv6—IPv6 Neighbor Discovery	12.0(22)S 12.2(14)S 12.2(28)SB 12.2(25)SG 12.2(33)SRA 12.2(17a)SX1 12.2(2)T 12.3 12.3(2)T 12.4 12.4(2)T 15.0(1)S	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 track neighboring routers.

Feature Name	Releases	Feature Information
IPv6—IPv6 Neighbor Discovery Duplicate Address Detection	12.0(22)S 12.2(14)S 12.2(28)SB 12.2(25)SG 12.2(33)SRA 12.2(17a)SX1 12.2(4)T 12.3 12.3(2)T 12.4 12.4(2)T	IPv6 neighbor discovery 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).
IPv6—IPv6 Stateless Autoconfiguration	12.0(22)S 12.2(14)S 12.2(28)SB 12.2(25)SG 12.2(33)SRA 12.2(17a)SX1 12.2(2)T 12.3 12.3(2)T 12.4 12.4(2)T 15.0(1)S	The IPv6 Stateless Autoconfiguration feature can be used to manage link, subnet, and site addressing changes.
IPv6—IPv6 Static Cache Entry for Neighbor Discovery	12.0(22)S 12.2(14)S 12.2(28)SB 12.2(33)SRA 12.2(17a)SX1 12.2(8)T 12.3 12.3(2)T 12.4 12.4(2)T 15.0(1)S	The IIPv6 Static Cache Entry for Neighbor Discovery feature allows static entries to be made in the IPv6 neighbor cache.

Feature Name	Releases	Feature Information
IPv6—Per-Interface Neighbor Discovery Cache Limit	15.1(3)T	The Per-Interface Neighbor Discovery Cache Limit feature provides the ability to limit the number of neighbor discovery cache entries on a per interface basis. The following commands were introduced or modified for this feature: **ipv6 nd cache interface-limit (global)**, **ipv6 nd cache interface-limit (interface)**, **show ipv6 neighbors**.
IPv6 Access Services: Routed Bridged Encapsulation (RBE)	12.3(4)T 12.4 12.4(2)T	RBE provides a mechanism for routing a protocol from a bridged interface to another routed or bridged interface.
IPv6 Address Types—Unicast	12.0(22)S 12.2(14)S 12.2(28)SB 12.2(33)SRA 12.2(17a)SX1 12.2(2)T 12.3 12.3(2)T 12.4 12.4(2)T	An IPv6 unicast address is an identifier for a single interface, on a single node.
IPv6 Data Link—ATM PVC and ATM LANE	12.0(22)S 12.2(14)S 12.2(28)SB 12.2(33)SRA 12.2(2)T 12.3 12.3(2)T 12.4 12.4(2)T	In IPv6 networks, a data link is a network sharing a particular link-local prefix. ATM PVC and ATM LANE are data links supported for IPv6.

Feature Name	Releases	Feature Information
IPv6 Data Link—Cisco High-Level Data Link Control (HDLC)	12.0(22)S 12.2(14)S 12.2(28)SB 12.2(33)SRA 12.2(2)T 12.3 12.3(2)T 12.4 12.4(2)T	In IPv6 networks, a data link is a network sharing a particular link-local prefix. HDLC is a type of data link supported for IPv6.
IPv6 Data Link—Dynamic Packet Transport (DPT)	12.0(23)S	In IPv6 networks, a data link is a network sharing a particular link-local prefix. DPT is a type of data link supported for IPv6.
IPv6 Data Link—Ethernet, Fast Ethernet, Gigabit Ethernet, and 10-Gigabit Ethernet	12.0(22)S 12.2(14)S 12.2(28)SB 12.2(33)SRA 12.2(2)T 12.3 12.3(2)T 12.4 12.4(2)T	In IPv6 networks, a data link is a network sharing a particular link-local prefix. Ethernet, Fast Ethernet, Gigabit Ethernet, and 10-Gigabit Ethernet are data links supported for IPv6.
IPv6 Data Link—FDDI	12.2(14)S 12.2(28)SB 12.2(33)SRA 12.2(2)T 12.3 12.3(2)T 12.4 12.4(2)T	In IPv6 networks, a data link is a network sharing a particular link-local prefix. FDDI is a type of data link supported for IPv6.

Feature Name	Releases	Feature Information
IPv6 Data Link—Frame Relay PVC	12.0(22)S 12.2(14)S 12.2(28)SB 12.2(33)SRA 12.2(2)T 12.3 12.3(2)T 12.4 12.4(2)T	In IPv6 networks, a data link is a network sharing a particular link-local prefix. Frame relay PVC is a type of data link supported for IPv6.
IPv6 Data Link—PPP service over Packet over SONET, ISDN, and serial (synchronous and asynchronous) interfaces	12.0(22)S 12.2(14)S 12.2(28)SB 12.2(33)SRA 12.2(2)T 12.3 12.3(2)T 12.4 12.4(2)T	In IPv6 networks, a data link is a network sharing a particular link-local prefix. PPP service over Packet over SONET, ISDN, and serial interfaces is a type of data link supported for IPv6.
IPv6 Data Link—VLANs using Cisco Inter-Switch Link (ISL)	12.0(22)S 12.2(14)S 12.2(28)SB 12.2(25)SG 12.2(33)SRA 12.2(18)SXE 12.2(2)T 12.3 12.3(2)T 12.4 12.4(2)T 15.0(1)S	In IPv6 networks, a data link is a network sharing a particular link-local prefix. VLANs using Cisco ISL is a type of data link supported for IPv6.

Feature Name	Releases	Feature Information
IPv6 Data Link—VLANs using IEEE 802.1Q encapsulation	12.0(22)S 12.2(28)SB 12.2(25)SG 12.2(33)SRA 12.2(18)SXE 12.2(14)S 12.2(2)T 12.3 12.3(2)T 12.4 12.4(2)T 15.0(1)S	In IPv6 networks, a data link is a network sharing a particular link-local prefix. VLANs using IEEE 802.1Q encapsulation is a type of data link supported for IPv6.
Enhanced IPv6 Neighbor Discovery Cache Management	12.2(33)SXI7 15.0(1)SY1	The IPv6 highly scalable neighbor discovery feature optimizes IPv6 neighbor discovery by providing ND cache autorefresh, unsolicited NA gleaning, and NUD exponential retransmit.
IPv6 Services—AAAA DNS lookups over an IPv4 Transport	12.0(22)S 12.2(14)S 12.2(28)SB 12.2(25)SG 12.2(33)SRA 12.2(17a)SX1 12.2(2)T 12.3 12.3(2)T 12.4 12.4(2)T 15.0(1)S	IPv6 basic connectivity can be enhanced by configuring support for AAAA record types in the DNS name-to-address and address-to-name lookup processes.

Feature Name	Releases	Feature Information
IPv6 Services—Cisco Discovery Protocol--IPv6 Address Family Support for Neighbor Information	12.2(14)S 12.2(28)SB 12.2(25)SG 12.2(33)SRA 12.2(18)SXE 12.2(8)T 12.3 12.3(2)T 12.4 12.4(2)T 15.0(1)S	The Cisco Discovery Protocol IPv6 Address Support for Neighbor Information feature adds the ability to transfer IPv6 addressing information between two Cisco devices.
IPv6 Services—DNS Lookups over an IPv6 Transport	12.0(22)S 12.2(14)S 12.2(28)SB 12.2(25)SG 12.2(33)SRE2 12.2(8)T 12.3 12.3(2)T 12.4 12.4(2)T 15.0(1)S	IPv6 supports DNS record types that are supported in the DNS name-to-address and address-to-name lookup processes.
IPv6 Services—Generic Prefix	12.3(4)T 12.4 12.4(2)T	The upper 64 bits of an IPv6 address are composed from a global routing prefix plus a subnet ID. A general prefix (for example, /48) holds a short prefix, based on which a number of longer, more specific, prefixes (for example, /64) can be defined.

Feature Name	Releases	Feature Information
IPv6 Switching—Cisco Express Forwarding and Distributed Cisco Express Forwarding Support	12.0(21)ST 12.0(22)S 12.2(14)S 12.2(28)SB 12.2(25)SG 12.2(33)SRA 12.2(17a)SX1 12.2(13)T 12.3 12.3(2)T 12.4 12.4(2)T 15.0(1)S	Cisco Express Forwarding for IPv6 is advanced, Layer 3 IP switching technology for the forwarding of IPv6 packets. Distributed Cisco Express Forwarding for IPv6 performs the same functions as CEFv6 but for distributed architecture platforms such as the GSRs and the Cisco 7500 series routers.
IPv6 Support on BVI Interfaces	15.1(2)T	This feature allows IPv6 commands to be supported on BVI so that users can assign IPv6 addresses to a BVI and route IPv6 packets.
Unicast Reverse Path Forwarding for IPv6	12.0(31)S 12.2(50)SY	The Unicast RPF feature mitigates problems caused by malformed or forged (spoofed) IPv6 source addresses that pass through an IPv6 router. The following command was introduced: **ipv6 verify unicast source reachable-via**.

Implementing ADSL for IPv6

This module describes the implementation of prefix pools, the authorization, authentication, and accounting (AAA) server, and per-user Remote Access Dial-In User Service (RADIUS) attributes in IPv6. It also describes the deployment of IPv6 in Digital Subscriber Line (DSL) and dial-access environments. Asymmetric Digital Subscriber Line (ADSL) provides the extensions that make large-scale access possible for IPv6 environments, including IPv6 RADIUS attributes, stateless address configuration on Point-to-Point Protocol (PPP) links, per-user static routes, and access control lists (ACLs).

Finding Feature Information

Your software release may not support all the features documented in this module. For the latest caveats and feature information, see Bug Search Tool and the release notes for your platform and software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the feature information table at the end of this module.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Restrictions for Implementing ADSL for IPv6

ADSL deployment is available for interfaces with PPP encapsulation enabled, including PPP over ATM (PPPoA), PPP over Ethernet (PPPoE, PPPoEoVLAN, PPPoEoQinQ) and PPPoEoA.

Information About Implementing ADSL for IPv6

Address Assignment for IPv6

A Cisco router configured with IPv6 will advertise its IPv6 prefixes on one or more interfaces, allowing IPv6 clients to automatically configure their addresses. In IPv6, address assignment is performed at the network layer, in contrast to IPv4 where a number of functions are handled in the PPP layer. The only function handled in IPv6 Control Protocol is the negotiation of a unique interface identifier. Everything else, including DNS server discovery, is done within the IPv6 protocol itself.

In IPv6, ISPs assign long-lived prefixes to users, which has some impact on the routing system. In typical IPv4 environments, each network access server (NAS) has a pool of 24-bit addresses and users get addresses from this pool when dialing in. If a user dials another POP or is connected to another NAS at the same POP, a different IPv4 address is assigned.

Addresses for IPv6 are assigned by the following methods.

- Stateless Address Autoconfiguration, page 72
- Prefix Delegation, page 72

Stateless Address Autoconfiguration

Assigning addresses using the stateless address autoconfiguration method can be used only to assign 64-bit prefixes. Each user is assigned a 64-bit prefix, which is advertised to the user in a router advertisement (RA). All addresses are automatically configured based on the assigned prefix.

A typical scenario is to assign a separate 64-bit prefix per user; however, users can also be assigned a prefix from a shared pool of addresses. Using the shared pool limits addresses to only one address per user.

This method works best for the cases where the customer provider edge (CPE) router is a single PC or is limited to only one subnet. If the user has multiple subnets, Layer 2 (L2) bridging, multilink subnets or proxy RA can be used. The prefix advertised in the RA can come from an authorization, authentication, and accounting (AAA) server, which also provides the prefix attribute, can be manually configured, or can be allocated from a prefix pool.

The Framed-Interface-Id AAA attribute influences the choice of interface identifier for peers and, in combination with the prefix, the complete IPv6 address can be determined.

Prefix Delegation

An IPv6 prefix delegating router selects IPv6 prefixes to be assigned to a requesting router upon receiving a request from the client. The delegating router might select prefixes for a requesting router in the following ways:

- Static assignment based on subscription to an ISP
- Dynamic assignment from a pool of available prefixes
- Selection based on an external authority such as a RADIUS server using the Delegated-IPv6-Prefix attribute

Contrary to IPv4 address assignment, an IPv6 user will be assigned a prefix, not a single address. Typically the Internet service provider (ISP) assigns a 64- or 48-bit prefix.

- Accounting Start and Stop Messages, page 73
- Forced Release of a Binding, page 73
- DHCP SIP Server Options, page 73

Accounting Start and Stop Messages

PPP calls a registry to allow DHCPv6 to append the delegated prefix information to accounting start and stop messages.

Forced Release of a Binding

The DHCPv6 server maintains an automatic binding table in memory to track the assignment of some configuration parameters, such as prefixes between the server and its clients. The automatic bindings can be stored permanently in the database agent, which can be, for example, a remote TFTP server or local NVRAM file system.

DHCPv6 invokes a routine when the virtual interface used by PPP terminates. This routine automatically releases any delegated prefix bindings associated with the PPP virtual interface that is being terminated.

When a PPP virtual interface terminates, the routine runs through the full table of DHCPv6 bindings checking for the matching interface. Because PPP uses a virtual interface, this subroutine clears any related lease information when the PPP connection terminates.

DHCP SIP Server Options

Two DHCP for IPv6 Session Initiation Protocol (SIP) server options describe a local outbound SIP proxy: one carries a list of domain names, the other a list of IPv6 addresses. These two options can be configured in a DHCPv6 configuration pool.

AAA over IPv6

Vendor-specific attributes (VSAs) are used to support AAA over IPv6. Cisco VSAs are inacl, outacl, prefix, and route.

You can configure prefix pools and pool names by using the AAA protocol. Customers can deploy an IPv6 RADIUS server or a TACACS+ server to communicate with Cisco devices.

- RADIUS over IPv6, page 73
- TACACS+ Over an IPv6 Transport, page 78
- IPv6 Prefix Pools, page 78

RADIUS over IPv6

The following RADIUS attributes, as described in RFC 3162, are supported for IPv6:

- Framed-Interface-Id
- Framed-IPv6-Pool
- Framed-IPv6-Prefix
- Framed-IPv6-Route
- Login-IPv6-Host

The following RADIUS attributes are also supported for IPv6:

- Delegated-IPv6-Prefix (RFC 4818)
- Delegated-IPv6-Prefix-Pool
- DNS-Server-IPv6-Address
- IPv6 ACL
- IPv6_DNS_Servers

- IPv6 Pool
- IPv6 Prefix#
- IPv6 Route

The attributes listed above can be configured on a RADIUS server and downloaded to access servers, where they can be applied to access connections.

Prerequisites for Using AAA Attributes for IPv6

AAA attributes for IPv6 are compliant with RFC 3162 and require a RADIUS server capable of supporting RFC 3162.

RADIUS Per-User Attributes for Virtual Access in IPv6 Environments

The following IPv6 RADIUS attributes are supported for virtual access and can be used as attribute-value (AV) pairs:

- Delegated-IPv6-Prefix
- Delegated-IPv6-Prefix-Pool
- DNS-Server-IPv6-Address
- Framed-Interface-Id
- Framed-IPv6-Pool
- Framed-IPv6-Prefix
- Framed-IPv6-Route
- IPv6 ACL
- IPv6_DNS_Servers
- IPv6 Pool
- IPv6 Prefix#
- IPv6 Route
- Login-IPv6-Host

Delegated-IPv6-Prefix

The Delegated-IPv6-Prefix attribute indicates an IPv6 prefix to be delegated to a user for use in a network. This attribute is used during DHCP prefix delegation between a RADIUS server and a delegating device. A Network Access Server (NAS) that hosts a DHCP Version 6 (DHCPv6) server can act as a delegating device.

The following example shows how to use the Delegated-IPv6-Prefix attribute:

```
ipv6:delegated-prefix=2001:DB8::/64
```

Note The Cisco VSA format is not supported for this attribute. If you try to add this attribute in the Cisco VSA format into a user profile, the RADIUS server response fails. Use only the IETF attribute format for this attribute.

Delegated-IPv6-Prefix-Pool

The Delegated-IPv6-Prefix-Pool attribute indicates the name of a prefix pool from which a prefix is selected and delegated to a device.

Prefix delegation is a DHCPv6 option for delegating IPv6 prefixes. Prefix delegation involves a delegating device that selects a prefix and assigns it on a temporary basis to a requesting device. A delegating device uses many strategies to choose a prefix. One method is to choose a prefix from a prefix pool with a name that is defined locally on a device.

The Delegated-IPv6-Prefix-Pool attribute indicates the name of an assigned prefix pool. A RADIUS server uses this attribute to communicate the name of a prefix pool to a NAS hosting a DHCPv6 server and acting as a delegating device.

You may use DHCPv6 prefix delegation along with ICMPv6 stateless address autoconfiguration (SLAAC) on a network. In this case, both the Delegated-IPv6-Prefix-Pool attribute and the Framed-IPv6-Pool attribute may be included within the same packet. To avoid ambiguity, the Delegated-IPv6-Prefix-Pool attribute should be restricted to the authorization and accounting of prefix pools used in DHCPv6 delegation, and the Framed-IPv6-Pool attribute should be used for the authorization and accounting of prefix pools used in SLAAC.

The following example shows how an address prefix is selected from a pool named pool1. The prefix pool pool1 is downloaded to a delegating device from a RADIUS server by using the Delegated-IPv6-Prefix-Pool attribute. The device then selects the address prefix 2001:DB8::/64 from this prefix pool.

```
Cisco:Cisco-AVpair = “ipv6:delegated-ipv6-pool = pool1”
!
ipv6 dhcp pool pool1
address prefix 2001:DB8::/64
!
```

DNS-Server-IPv6-Address

The DNS-Server-IPv6-Address attribute indicates the IPv6 address of a Domain Name System (DNS) server. A DHCPv6 server can configure a host with the IPv6 address of a DNS server. The IPv6 address of the DNS server can also be conveyed to the host using router advertisement messages from ICMPv6 devices.

A NAS may host a DHCPv6 server to handle DHCPv6 requests from hosts. The NAS may also act as a device that provides router advertisement messages. Therefore, this attribute is used to provide the NAS with the IPv6 address of the DNS server.

If a NAS has to announce more than one recursive DNS server to a host, this attribute can be included multiple times in Access-Accept packets sent from the NAS to the host.

The following example shows how you can define the IPv6 address of a DNS server by using the DNS-Server-IPv6-Address attribute:

```
Cisco:Cisco-AVpair = "ipv6:ipv6-dns-servers-addr=2001:DB8::"
```

Framed-Interface-Id

The Framed-Interface-Id attribute indicates an IPv6 interface identifier to be configured for a user.

This attribute is used during IPv6 Control Protocol (IPv6CP) negotiations of the Interface-Identifier option. If negotiations are successful, the NAS uses this attribute to communicate a preferred IPv6 interface identifier to the RADIUS server by using Access-Request packets. This attribute may also be used in Access-Accept packets.

Framed-IPv6-Pool

The Framed-IPv6-Pool attribute indicates the name of a pool that is used to assign an IPv6 prefix to a user. This pool should be either defined locally on a device or defined on a RADIUS server from where pools can be downloaded.

Framed-IPv6-Prefix

The Framed-IPv6-Prefix attribute indicates an IPv6 prefix (and a corresponding route) to be configured for a user. So this attribute performs the same function as a Cisco VSA and is used for virtual access only. A NAS uses this attribute to communicate a preferred IPv6 prefix to a RADIUS server by using Access-Request packets. This attribute may also be used in Access-Accept packets and can appear multiple times in these packets. The NAS creates a corresponding route for the prefix.

This attribute is used by a user to specify which prefixes to advertise in router advertisement messages of the Neighbor Discovery Protocol.

This attribute can also be used for DHCPv6 prefix delegation, and a separate profile must be created for a user on the RADIUS server. The username associated with this separate profile has the suffix “-dhcpv6”.

The Framed-IPv6-Prefix attribute is treated differently in this separate profile and the regular profile of a user. If a NAS needs to send a prefix through router advertisement messages, the prefix is placed in the Framed-IPv6-Prefix attribute of the regular profile of the user. If a NAS needs to delegate a prefix to the network of a remote user, the prefix is placed in the Framed-IPv6-Prefix attribute of the separate profile of the user.

Note The RADIUS IETF attribute format and the Cisco VSA format are supported for this attribute.

Framed-IPv6-Route

The Framed-IPv6-Route attribute indicates the routing information to be configured for a user on a NAS. This attribute performs the same function as a Cisco VSA. The value of the attribute is a string and is specified by using the **ipv6 route** command.

IPv6 ACL

The IPv6 ACL attribute is used to specify a complete IPv6 access list. The unique name of an access list is generated automatically. An access list is removed when the respective user logs out. The previous access list on the interface is then reapplied.

The inacl and outacl attributes enable you to specify an existing access list configured on a device. The following example shows how to define an access list identified with number 1:

```
cisco-avpair = "ipv6:inacl#1=permit 2001:DB8:cc00:1::/48",
cisco-avpair = "ipv6:outacl#1=deny 2001:DB8::/10",
```

IPv6_DNS_Servers

The IPv6_DNS_Servers attribute is used to send up to two DNS server addresses to the DHCPv6 server. The DNS server addresses are saved in the interface DHCPv6 subblock and override other configurations in the DHCPv6 pool. This attribute is also included in attributes returned for AAA start and stop notifications.

IPv6 Pool

The IPv6 Pool attribute extends the IPv4 address pool attribute to support the IPv6 protocol for RADIUS authentication. This attribute specifies the name of a local pool on a NAS from which a prefix is chosen and used whenever PPP is configured and the protocol is specified as IPv6. The address pool works with local pooling and specifies the name of a local pool that is preconfigured on the NAS.

IPv6 Prefix#

The IPv6 Prefix# attribute indicates which prefixes to advertise in router advertisement messages of the Neighbor Discovery Protocol. When this attribute is used, a corresponding route (marked as a per-user static route) is installed in the routing information base (RIB) tables for a given prefix.

The following example shows how to specify which prefixes to advertise:

```
cisco-avpair = "ipv6:prefix#1=2001:DB8::/64",
cisco-avpair = "ipv6:prefix#2=2001:DB8::/64",
```

IPv6 Route

The IPv6 Route attribute is used to specify a static route for a user. A static route is appropriate when Cisco software cannot dynamically build a route to the destination. See the **ipv6 route** command for more information about building static routes.

The following example shows how to use the IPv6 Route attribute to define a static route:

```
cisco-avpair = "ipv6:route#1=2001:DB8:cc00:1::/48",
cisco-avpair = "ipv6:route#2=2001:DB8:cc00:2::/48",
```

Login-IPv6-Host

The Login-IPv6-Host attribute indicates IPv6 addresses of hosts with which to connect a user when the Login-Service attribute is included. A NAS uses the Login-IPv6-Host attribute in Access-Request packets to communicate to a RADIUS server that it prefers to use certain hosts.

PPP IPv6 Accounting Delay Enhancements

This feature enhances accounting records for dual-stack networks. It ensures that a unique IPv6 address is assigned to PPP IPv6 and IPv4 sessions for IP addresses that are received from RADIUS.

When this feature is enabled, it automatically creates a database to hold new incoming access-accept responses from RADIUS. The access-accept responses in this database are then checked for duplicates of a specific set of attributes. If the attributes are already present in the database, then the RADIUS server has already offered them to an existing session; therefore, the new session is immediately removed and a stop-record message sent. If none of the specific set of attributes are in the database, they are immediately added to the database, and the session proceeds normally. When the session is removed, the entries in the database are also removed.

The following RADIUS attributes are tracked in the database and checked at access-accept time:

- Framed-IPv6-Prefix
- Delegated-IPv6-Prefix

The attributes are available as standard RFC-defined binary format, or as Cisco VSAs. (The Delegated-IPv6-Prefix attribute currently does not have a VSA definition in AAA.)

TACACS+ Over an IPv6 Transport

An IPv6 server can be configured to use TACACS+. Both IPv6 and IPv4 servers can be configured to use TACACS+ using a name instead of an IPv4 or IPv6 address.

IPv6 Prefix Pools

The function of prefix pools in IPv6 is similar to that of address pools in IPv4. The main difference is that IPv6 assigns prefixes rather than single addresses.

As in IPv4, a pool or a pool definition in IPv6 can be configured locally or it can be retrieved from an AAA server. Overlapping membership between pools is not permitted.

Once a pool is configured, it cannot be changed. If you change the configuration, the pool will be removed and re-created. All prefixes previously allocated will be freed.

Prefix pools can be defined so that each user is allocated a 64-bit prefix or so that a single prefix is shared among several users. In a shared prefix pool, each user may receive only one address from the pool.

Broadband IPv6 Counter Support at LNS

This feature provides support for broadband PPP IPv6 sessions at the layer 2 tunneling protocol (L2TP) network server (LNS). The sessions are forwarded by L2TP access concentrator (LAC) using layer 2 tunneling protocol L2TP over IPv6.

This feature is enabled automatically when the user configures LNS and enables IPv6.

How to Configure ADSL in IPv6

Configuring the NAS

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **hostname** *name*
4. **aaa new-model**
5. **aaa authentication ppp** {**default** | *list-name*} *method1* [*method2*...]
6. **aaa authorization configuration default** {**radius** | **tacacs+**
7. **show ipv6 route** [*ipv6-address* | *ipv6-prefix* / *prefix-length* | *protocol* | *interface-type interface-number*
8. **virtual-profile virtual-template** *number*
9. **interface serial** *controller-number* **:** *timeslot*
10. **encapsulation** *encapsulation-type*
11. **exit**
12. **dialer-group** *group-number*
13. **ppp authentication** *protocol1* [*protocol2*...] [**if-needed**] [*list-name* | **default**] [**callin**] [**one-time**] [**optional**]
14. **interface virtual-template** *number*
15. **ipv6 enable**
16. **dialer-list** *dialer-group* **protocol** *protocol-name* {**permit** | **deny** | **list** *access-list-number* | *access-group*}
17. **radius-server host** {*hostname* | *ip-address*} [**test username** *user-name*] [**auth-port** *port-number*] [**ignore-auth-port**] [**acct-port** *port-number*] [**ignore-acct-port**] [**timeout** *seconds*] [**retransmit** *retries*] [**key** *string*] [**alias** {*hostname* | *ip-address*}] [**idle-time** *seconds*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**hostname** *name* **Example:** `Router(config)# hostname cust1-53a`	Specifies the hostname for the network server.

	Command or Action	Purpose
Step 4	**aaa new-model** **Example:** `Router(config)# aaa new-model`	Enables the AAA server.
Step 5	**aaa authentication ppp** {**default** | *list-name*} *method1* [*method2*...] **Example:** `Router(config)# aaa authentication ppp default if-needed group radius`	Specifies one or more AAA authentication methods for use on serial interfaces that are running PPP.
Step 6	**aaa authorization configuration default** {**radius** | **tacacs+** **Example:** `Router(config)# aaa authorization configuration default radius`	Downloads configuration information from the AAA server.
Step 7	**show ipv6 route** [*ipv6-address* | *ipv6-prefix* / *prefix-length* | *protocol* | *interface-type interface-number* **Example:** `Router(config)# show ipv6 route`	Shows the routes installed by the previous commands.
Step 8	**virtual-profile virtual-template** *number* **Example:** `Router(config)# virtual-profile virtual-template 1`	Enables virtual profiles by virtual interface template.
Step 9	**interface serial** *controller-number* **:** *timeslot* **Example:** `Router(config)# interface serial 0:15`	Specifies a serial interface created on a channelized E1 or channelized T1 controller (for ISDN PRI, channel-associated signaling, or robbed-bit signaling). This command also puts the router into interface configuration mode.
Step 10	**encapsulation** *encapsulation-type* **Example:** `Router(config-if)# encapsulation ppp`	Sets the encapsulation method used by the interface.

	Command or Action	Purpose
Step 11	**exit** **Example:** `Router(config-if)# exit`	Returns to global configuration mode.
Step 12	**dialer-group** *group-number* **Example:** `Router(config)# dialer-group 1`	Controls access by configuring an interface to belong to a specific dialing group.
Step 13	**ppp authentication** *protocol1* [*protocol2...*] [**if-needed**] [*list-name* | **default**] [**callin**] [**one-time**] [**optional**] **Example:** `Router(config)# ppp authentication chap`	Enables Challenge Handshake Authentication Protocol (CHAP) or Password Authentication Protocol (PAP) or both and specifies the order in which CHAP and PAP authentication are selected on the interface.
Step 14	**interface virtual-template** *number* **Example:** `Router(config)# interface virtual-template 1`	Creates a virtual template interface that can be configured and applied dynamically in creating virtual access interfaces.
Step 15	**ipv6 enable** **Example:** `Router(config)# ipv6 enable`	Enables IPv6 processing on an interface that has not been configured with an explicit IPv6 address.
Step 16	**dialer-list** *dialer-group* **protocol** *protocol-name* {**permit** | **deny** | **list** *access-list-number* | *access-group*} **Example:** `Router(config)# dialer-list 1 protocol ipv6 permit`	Defines a dial-on-demand routing (DDR) dialer list for dialing by protocol or by a combination of a protocol and a previously defined access list.

	Command or Action	Purpose
Step 17	**radius-server host** {*hostname* | *ip-address*} [**test username** *username*] [**auth-port** *port-number*] [**ignore-auth-port**] [**acct-port** *port-number*] [**ignore-acct-port**] [**timeout** *seconds*] [**retransmit** *retries*] [**key** *string*] [**alias** {*hostname* | *ip-address*}] [**idle-time** *seconds* **Example:** `Router(config)# radius-server host 172.17.250.8 auth-port 1812 acct-port 1813 key testing123`	Specifies a RADIUS server host.

Enabling the Sending of Accounting Start and Stop Messages

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ipv6 dhcp pool** *poolname*
4. **accounting** *mlist*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**ipv6 dhcp pool** *poolname* **Example:** `Device(config)# ipv6 dhcp pool pool1`	Configures a DHCP for IPv6 configuration information pool and enters DHCP for IPv6 pool configuration mode.

	Command or Action	Purpose
Step 4	**accounting** *mlist* **Example:** `Device(config-dhcp)# accounting list1`	Enables accounting start and stop messages to be sent.

Forcing Release of Prefix Bindings

Perform this task to release any delegated prefix bindings associated with the PPP virtual interface that is being terminated.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type number*
4. **ipv6 dhcp bindings track ppp**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *type number* **Example:** `Device(config)# interface VirtualAccess2.2`	Specifies an interface type and number, and places the router in interface configuration mode.
Step 4	**ipv6 dhcp bindings track ppp** **Example:** `Device(config-if)# ipv6 dhcp bindings track ppp`	Releases any delegated prefix leases associated with the PPP virtual interface that is being terminated.

Configuring DHCPv6 AAA Options

Perform the following task to configure the option of acquiring prefixes from the AAA server:

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ipv6 dhcp pool** *pool-name*
4. **prefix-delegation aaa** [**method-list** *method-list*] [*lifetime*]
5. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**ipv6 dhcp pool** *pool-name* **Example:** `Device(config)# ipv6 dhcp pool pool1`	Configures a DHCPv6 configuration information pool and enters IPv6 DHCP pool configuration mode.
Step 4	**prefix-delegation aaa** [**method-list** *method-list*] [*lifetime*] **Example:** `Device(config-dhcpv6)# prefix-delegation aaa method-list list1`	Specifies that prefixes are to be acquired from AAA servers.
Step 5	**end** **Example:** `Device(config-dhcpv6)# end`	Exits IPv6 DHCP pool configuration mode and returns to privileged EXEC mode.

Configuring PPP IPv6 Accounting Delay Enhancements

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ppp unique address access-accept**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**ppp unique address access-accept** **Example:** `Router(config)# ppp unique address access-accept`	Tracks duplicate addresses received from RADIUS and creates a standalone database.

Configuring TACACS+ over IPv6

Configuring the TACACS+ Server over IPv6

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **tacacs server** *name*
4. **address ipv6** *ipv6-address*
5. **key** [**0** | **7**] *key-string*
6. **port** [*number*
7. **send-nat-address**
8. **single-connection**
9. **timeout** *seconds*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**tacacs server** *name* **Example:** `Router(config)# tacacs server server1`	Configures the TACACS+ server for IPv6 and enters TACACS+ server configuration mode.
Step 4	**address ipv6** *ipv6-address* **Example:** `Router(config-server-tacacs)# address ipv6 2001:DB8:3333:4::5`	Configures the IPv6 address of the TACACS+ server.
Step 5	**key** [**0**	**7**] *key-string* **Example:** `Router(config-server-tacacs)# key 0 key1`	Configures the per-server encryption key on the TACACS+ server.

	Command or Action	Purpose
Step 6	**port** [*number* **Example:** `Router(config-server-tacacs)# port 12`	Specifies the TCP port to be used for TACACS+ connections.
Step 7	**send-nat-address** **Example:** `Router(config-server-tacacs)# send-nat-address`	Sends a client's post-NAT address to the TACACS+ server.
Step 8	**single-connection** **Example:** `Router(config-server-tacacs)# single-connection`	Enables all TACACS packets to be sent to the same server using a single TCP connection.
Step 9	**timeout** *seconds* **Example:** `Router(config-server-tacacs)# timeout 10`	Configures the time to wait for a reply from the specified TACACS server.

Specifying the Source Address in TACACS+ Packets

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ipv6 tacacs source-interface** *type number*

DETAILED STEPS

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

Command or Action	Purpose
Step 2 **configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3 **ipv6 tacacs source-interface** *type number* **Example:** `Router(config)# ipv6 tacacs source-interface GigabitEthernet 0/0/0`	Specifies an interface to use for the source address in TACACS+ packets.

Configuring TACACS+ Server Group Options

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **aaa group server tacacs+** *group-name*
4. **server name** *server-name*
5. **server-private** {*ip-address* | *name* | *ipv6-address*} [**nat**] [**single-connection**] [**port** *port-number*] [**timeout** *seconds*] [**key** [**0** | **7**] *string*]

DETAILED STEPS

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

	Command or Action	Purpose
Step 3	**aaa group server tacacs+** *group-name* **Example:** `Router(config)# aaa group server tacacs+ group1`	Groups different TACACS+ server hosts into distinct lists and distinct methods.
Step 4	**server name** *server-name* **Example:** `Router(config-sg-tacacs+)# server name server1`	Specifies an IPv6 TACACS+ server.
Step 5	**server-private** {*ip-address* | *name* | *ipv6-address*} [**nat**] [**single-connection**] [**port** *port-number*] [**timeout** *seconds*] [**key** [**0** | **7**] *string*] **Example:** `Router(config-sg-tacacs+)# server-private 2001:DB8:3333:4::5 port 19 key key1`	Configures the IPv6 address of the private TACACS+ server for the group server.

Verifying Broadband IPv6 Counter Support at the LNS

This feature is enabled automatically when the user configures LNS and enables IPv6. To verify information about this feature, you can use any or all of the following optional commands as needed.

SUMMARY STEPS

1. **enable**
2. **show l2tp session** [**all** | **packets** [**ipv6**] | **sequence** | **state** | [**brief** | **circuit** | **interworking**] [**hostname**]] [**ip-addr** *ip-addr*[**vcid** *vcid*] | **tunnel**{**id** *local-tunnel-id local-session-id*| **remote-name** *remote-tunnel-name local-tunnel-name*}| **username** *username* | **vcid** *vcid*]
3. **show l2tp tunnel** [**all** | **packets** [**ipv6**] | **state** | **summary** | **transport**] [**id** *local-tunnel-id* | **local-name** *local-tunnel-name remote-tunnel-name*| **remote-name** *remote-tunnel-name local-tunnel-name*]
4. **show l2tun session** [**l2tp** | **pptp**] [**all** [*filter*] | **brief** [*filter*] [**hostname**] | **circuit** [*filter*] [**hostname**] | **interworking** [*filter*] [**hostname**] | **packets ipv6**] [*filter*] | **sequence** [*filter*] | **state** [*filter*]]
5. **show vpdn session** [**l2f** | **l2tp** | **pptp**] [**all** | **packets** [**ipv6**] | **sequence** | **state** [*filter*]]
6. **show vpdn tunnel** [**l2f** | **l2tp** | **pptp**] [**all** [*filter*] | **packets ipv6**] [*filter*] | **state** [*filter*] | **summary** [*filter*] | **transport**[*filter*]]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**show l2tp session** [**all** | **packets** [**ipv6**] | **sequence** | **state** | [**brief** | **circuit** | **interworking**] [**hostname**]] [**ip-addr** *ip-addr*[**vcid** *vcid*] | **tunnel**{**id** *local-tunnel-id local-session-id*| **remote-name** *remote-tunnel-name local-tunnel-name*}| **username** *username* | **vcid** *vcid*] **Example:** `Router# show l2tp session packets ipv6`	Displays information about L2TP sessions.
Step 3	**show l2tp tunnel** [**all** | **packets** [**ipv6**] | **state** | **summary** | **transport**] [**id** *local-tunnel-id* | **local-name** *local-tunnel-name remote-tunnel-name*| **remote-name** *remote-tunnel-name local-tunnel-name*] **Example:** `Router# show l2tp tunnel packets ipv6`	Displays details about L2TP tunnels.
Step 4	**show l2tun session** [**l2tp** | **pptp**] [**all** [*filter*] | **brief** [*filter*] [**hostname**] | **circuit** [*filter*] [**hostname**] | **interworking** [*filter*] [**hostname**] | **packets ipv6**] [*filter*] | **sequence** [*filter*] | **state** [*filter*]] **Example:** `Router# show l2tun session packets ipv6`	Displays the current state of Layer 2 sessions and protocol information about L2TP control channels.
Step 5	**show vpdn session** [**l2f** | **l2tp** | **pptp**] [**all** | **packets** [**ipv6**] | **sequence** | **state** [*filter*]] **Example:** `Router# show vpdn session packets ipv6`	Displays session information about active Layer 2 sessions for a virtual private dialup network (VPDN).
Step 6	**show vpdn tunnel** [**l2f** | **l2tp** | **pptp**] [**all** [*filter*] | **packets ipv6**] [*filter*] | **state** [*filter*] | **summary** [*filter*] | **transport**[*filter*]] **Example:** `Router# show vpdn tunnel packets ipv6`	Displays information about active Layer 2 tunnels for a VPDN.

Configuration Examples for Implementing ADSL for IPv6

Example NAS Configuration

This configuration for the ISP NAS shows the configuration that supports access from the remote CE router.

```
hostname hostname1
  aaa new-model
  aaa authentication ppp default if-needed group radius
  aaa authorization network default

  aaa accounting network default start-stop group radius

  aaa accounting send counters ipv6

  interface virtual-template 1

  ip unnumbered loopback interface1

  ipv6 address autoconfig

  no ipv6 nd ra suppress
  ppp authentication chap

  ppp accounting list1

  no snmp trap link-status

  no logging event link-status

  exit

  aaa group service radius group1

  server-private 10.1.1.1 timeout 5 retransmit 3 key xyz

  radius-server host 192.0.2.176 test username test1 auth-port 1645 acct-port 1646

  radius-server vsa send accounting

  radius-server vsa send authentication
```

Example RADIUS Configuration

This RADIUS configuration shows the definition of AV pairs to establish the static routes.

```
  campus1 Auth-Type = Local, Password = "mypassword"
               User-Service-Type = Framed-User,
               Framed-Protocol = PPP,
```

```
cisco-avpair = "ipv6:inacl#1=permit dead::/64 any",
cisco-avpair = "ipv6:route=library::/64",
cisco-avpair = "ipv6:route=cafe::/64",
cisco-avpair = "ipv6:prefix=library::/64 0 0 onlink autoconfig",
cisco-avpair = "ipv6:prefix=cafe::/64 0 0 onlink autoconfig",
cisco-avpair = "ip:route=10.0.0.0 255.0.0.0",
```

Examples: Verifying Broadband IPv6 Counter Support at the LNS

Example: show l2tp session Command

The **show l2tp session** command used with the **packets and ipv6**keywords displays information about IPv6 packets and byte counts in an L2TP session.

```
Router# show l2tp session packets ipv6


L2TP Session Information Total tunnels 1 sessions 1


LocID      RemID      TunID      Pkts-In    Pkts-Out   Bytes-In   Bytes-Out

16791      53352      27723      30301740   30301742   20159754280 20523375360
```

Example: show l2tp tunnel Command

The **show l2tp tunnel** command used with the **packets** and **ipv6** keywords displays information about IPv6 packet statistics and byte counts in L2TP tunnels.

```
Router# show l2tp tunnel packets ipv6
L2TP Tunnel Information Total tunnels 1 sessions 1
LocTunID   Pkts-In    Pkts-Out   Bytes-In   Bytes-Out
27723      63060379   63060383   39400320490 40157045438
```

Example: show l2tun session Command

The **show l2tun session** command used with the **packets**and **ipv6**keywords displays information about IPv6 packet statistics and byte counts in an L2TUN session.

```
Router# show l2tun session packets ipv6
L2TP Session Information Total tunnels 1 sessions 1
LocID      RemID      TunID      Pkts-In    Pkts-Out   Bytes-In   Bytes-Out
16791      53352      27723      31120707   31120708   21285014938 21658462236
```

Example: show vpdn session Command

The **show vpdn session** command used with the **l2tp**, **packets**, and **ipv6**keywords displays session information about IPv6 packet statistics and byte counts in an active layer 2 session for a VPDN.

```
Router# show vpdn session l2tp packets ipv6
L2TP Session Information Total tunnels 1 sessions 1
LocID      RemID      TunID      Pkts-In    Pkts-Out   Bytes-In   Bytes-Out
16791      53352      27723      35215536   35215538   22616342688 23038929320
```

Example: show vpdn tunnel Command

The **show vpdn tunnel**command used with the **l2tp**, **packets**, and **ipv6**keywords displays session information about IPv6 packet statistics and byte counts in an active layer 2 tunnel for a VPDN.

```
Device# show vpdn tunnel l2tp packets ipv6
L2TP Tunnel Information Total tunnels 1 sessions 1
LocTunID   Pkts-In    Pkts-Out   Bytes-In   Bytes-Out
27723      61422447   61422451   37149801922 37886871686
```

Additional References

Related Documents

Related Topic	Document Title
IPv6 supported feature list	" Start Here: Cisco IOS XE Software Release Specifics for IPv6 Features ," *Cisco IOS XE IPv6 Configuration Guide*
IPv6 basic connectivity	" Implementing IPv6 Addressing and Basic Connectivity, " *Cisco IOS XE IPv6 Configuration Guide*
DHCP for IPv6	" Implementing DHCP for IPv6, " *Cisco IOS XE IPv6 Configuration Guide*
IPv6 commands: complete command syntax, command mode, command history, defaults, usage guidelines, and examples	*Cisco IOS IPv6 Command Reference*

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
None	To locate and download MIBs for selected platforms, Cisco software releases, and feature sets, use Cisco MIB Locator found at the following URL: http://www.cisco.com/go/mibs

RFCs

RFCs	Title
RFC 3162	*RADIUS and IPv6*
RFC 3177	*IAB/IESG Recommendations on IPv6 Address*
RFC 3319	*Dynamic Host Configuration Protocol (DHCPv6) Options for Session Initiated Protocol (SIP) Servers*

Technical Assistance

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

Feature Information for Implementing ADSL for IPv6

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Table 6 Feature Information for Implementing ADSL for IPv6

Feature Name	Releases	Feature Information
Enhanced IPv6 Features for ADSL and Dial Deployment	Cisco IOS XE Release 2.5	Several features were enhanced to enable IPv6 to use ADSL and dial deployment.
AAA Support for Cisco VSA IPv6 Attributes	Cisco IOS XE Release 2.5	Vendor-specific attributes (VSAs) were developed to support AAA for IPv6.
IPv6 Access Services: PPPoE	Cisco IOS XE Release 2.5	ADSL and dial deployment is available for interfaces with PPP encapsulation enabled, including PPPoE.
AAA Support for RFC 3162 IPv6 RADIUS Attributes	Cisco IOS XE Release 2.5	The AAA attributes for IPv6 are compliant with RFC 3162 and require a RADIUS server capable of supporting RFC 3162. The following commands were modified by this feature: **ipv6 dhcp pool**, prefix-delegation aaa
DHCP - DHCPv6 Prefix Delegation RADIUS VSA	Cisco IOS XE Release 2.5	When the user requests a prefix from the prefix delegator, typically the NAS, the prefix is allocated using DHCPv6.
PPP Enhancement for Broadband IPv6	Cisco IOS XE Release 2.5	The following sections provide information about this feature.
AAA Improvements for Broadband IPv6	Cisco IOS XE Release 2.5	
DHCP Enhancements to Support IPv6 Broadband Deployments	Cisco IOS XE Release 2.5	
PPPoA	Cisco IOS XE Release 3.3S	ADSL and dial deployment is available for interfaces with PPP encapsulation enabled, including PPPoA.
SSO - PPPoE IPv6	Cisco IOS XE Release 2.5	This feature is supported in Cisco IOS XE Release 2.5.

Feature Name	Releases	Feature Information
Broadband IPv6 Counter Support at LNS	Cisco IOS XE Release 2.6	This feature provides support for broadband PPP IPv6 sessions at the L2TP LNS. The sessions are forwarded by LAC using layer 2 tunneling protocol L2TP over IPv4. The following commands were modified by this feature: **show l2tp session**, **show l2tp tunnel**, **show l2tun session**, **show vpdn session**, **show vpdn tunnel**.
PPP IPv6 Accounting Delay Enhancements	Cisco IOS XE Release 3.2S	This feature enhances accounting records for dual-stack networks. It ensures that a unique IPv6 address is assigned to PPP IPv6 and IPv4 sessions for IP addresses that are received from RADIUS. The following command was introduced by this feature: **debug ppp unique address**, **ppp unique address access-accept**
RADIUS over IPv6	Cisco IOS XE Release 3.2S	RADIUS over IPv6 is supported.
TACACS+ over IPv6	Cisco IOS XE Release 3.2S	TACACS+ over IPv6 is supported. The following commands were introduced or modified by this feature: **aaa group server tacacs +**, **address ipv6 (TACACS+)**, **ipv6 tacacs source-interface**, **key (TACACS+)**, **port (TACACS+)**, **send-nat-address**, **server name (IPv6 TACACS+)**, **server-private (TACACS+)**, **single-connection**, **tacacs server**, **timeout (TACACS+)**.

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.

Implementing Bidirectional Forwarding Detection for IPv6

This document describes how to implement the Bidirectional Forwarding Detection for IPv6 (BFDv6) protocol. BFD is a detection protocol designed to provide fast forwarding path failure detection times for all media types, encapsulations, topologies, and routing protocols. In addition to fast forwarding path failure detection, BFD provides a consistent failure detection method for network administrators. BFDv6 provides IPv6 support by accommodating IPv6 addresses, and it provides the ability to create BFDv6 sessions.

Because the network administrator can use BFD to detect forwarding path failures at a uniform rate, rather than the variable rates for different routing protocol hello mechanisms, network profiling and planning will be easier, and reconvergence time will be consistent and predictable.

Finding Feature Information

Your software release may not support all the features documented in this module. For the latest caveats and feature information, see Bug Search Tool and the release notes for your platform and software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the feature information table at the end of this module.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Prerequisites for Implementing Bidirectional Forwarding Detection for IPv6

IPv6 Cisco Express Forwarding and IPv6 unicast routing must be enabled on all participating routers.

Restrictions for Implementing Bidirectional Forwarding Detection for IPv6

- BFDv6 supports only global IPv6 neighbor addresses if a global IPv6 address is configured on the interface.
- Only asynchronous mode is supported. In asynchronous mode, either BFDv6 peer can initiate a BFDv6 session.

Information About Implementing Bidirectional Forwarding Detection for IPv6

Overview of the BFDv6 Protocol

This section describes the BFDv6 protocol, how it is different from BFD for IPv4, and how it works with BFD for IPv4. BFD is a detection protocol designed to provide fast forwarding path failure detection times for all media types, encapsulations, topologies, and routing protocols. In addition to fast forwarding path failure detection, BFD provides a consistent failure detection method for network administrators. BFDv6 provides IPv6 support by accommodating IPv6 addresses and provides the ability to create BFDv6 sessions.

BFDv6 Registration

BFD clients register with BFD using a registry application program interface (API). The registry arguments include protocol type and the address and interface description block (IDB) of the route to be monitored. These APIs and arguments are all assumed by BFD to be IPv4.

BFDv6 has registries from which these arguments have been removed, and the protocol and encapsulation are described within a session information structure. These session information structures are defined by BFDv6 for the protocols supported. BFDv6 uses information from the session information structures to determine the correct encapsulation for BFDv6 packets on that session.

BFDv6 Global and Link-Local Addresses

BFDv6 supports both global and link-local IPv6 addresses for neighbor creation. BFDv6 sessions select source addresses to match the neighbor address types (for example, global IPv6 address neighbors must be paired with global IPv6 source addresses and link-local IPv6 address neighbors must be paired with link-local IPv6 source addresses). The table below shows the address pairings that BFDv6 supports.

Table 7 BFDv6 Address Pairings for Neighbor Creation

Source Address	Destination Address	Status
Global	Global	Supported
Global	Link local	Not supported
Link local	Global	Not supported
Link local	Link local	Supported

Because all IPv6-enabled interfaces have a link-local address and BFDv6 selects the source address, link-local address neighbors are always paired with a link-local interface address. The link-local source address with global destination address is not supported by Cisco Express Forwarding. Therefore, a global IPv6 address must be configured on an interface before a session with a global address neighbor may be established in BFDv6. BFDv6 rejects any sessions in which the neighbor address is global and no global address is configured on the interface.

Note The behavior of a unique local address (ULA) in BFDv6 is the same as a global address.

BFD for IPv4 and IPv6 on the Same Interface

BFD supports multiple IPv4 and IPv6 sessions per interface, with no restriction on the protocol of those sessions.

Static Route Support for BFD over IPv6

Using the BFDv6 protocol to reach the static route next hop ensures that an IPv6 static route is inserted only in the IPv6 Routing Information Base (RIB) when the next-hop neighbor is reachable. Using the BFDv6 protocol also can remove the IPv6 static route from the IPv6 RIB when the next hop becomes unreachable.

You can configure IPv6 static BFDv6 neighbors. These neighbors can operate in one of two modes: associated (which is the default) and unassociated. A neighbor can be transitioned between the two modes without interrupting the BFDv6 session associated with the neighbor.

- BFDv6 Associated Mode, page 101
- BFDv6 Unassociated Mode, page 102

BFDv6 Associated Mode

In Bidirectional Forwarding Detection for IPv6 (BFDv6) associated mode, an IPv6 static route is automatically associated with an IPv6 static BFDv6 neighbor if the static route next hop exactly matches the static BFDv6 neighbor.

An IPv6 static route requests a BFDv6 session for each static BFDv6 neighbor that has one or more associated IPv6 static routes and is configured over an interface on which BFD has been configured. The state of the BFDv6 session will be used to determine whether the associated IPv6 static routes are inserted in the IPv6 RIB. For example, static routes are inserted in the IPv6 RIB only if the BFDv6 neighbor is

reachable, and the static route is removed from the IPv6 RIB if the BFDv6 neighbor subsequently becomes unreachable.

BFDv6 associated mode requires you to configure a BFD neighbor and static route on both the router on which the BFD-monitored static route is required and on the neighboring router.

BFDv6 Unassociated Mode

An IPv6 static BFD neighbor may be configured as unassociated. In this mode, the neighbor is not associated with static routes, and the neighbor always requests a BFDv6 session if the interface has been configured for BFDv6.

Unassociated mode is useful in the following situations:

- Bringing up a BFDv6 session in the absence of an IPv6 static route—This case occurs when a static route is on router A, with router B as the next hop. Associated mode requires you to create both a static BFD neighbor and static route on both routers in order to bring up the BFDv6 session from B to A. Specifying the static BFD neighbor in unassociated mode on router B avoids the need to configure an unwanted static route.
- Transition to BFD monitoring of a static route—This case occurs when existing IPv6 static routes are inserted in the IPv6 RIB. Here, you want to enable BFD monitoring for these static routes without any interruption to traffic. If you configure an attached IPv6 static BFD neighbor, then the static routes will immediately be associated with the new static BFD neighbor. However, because a static BFD neighbor starts in a down state, the associated static routes are then removed from the IPv6 RIB and are reinserted when the BFDv6 session comes up. Therefore, you will see an interruption in traffic. This interruption can be avoided by configuring the static BFD neighbor as unassociated, waiting until the BFDv6 session has come up, and then reconfiguring the static BFD neighbor as associated.
- Transition from BFD monitoring of a static route—In this case, IPv6 static routes are monitored by BFD and inserted in the RIB. Here, you want to disable BFD monitoring of the static routes without interrupting traffic flow. This scenario can be achieved by first reconfiguring the static BFD neighbor as detached (thus disassociating the neighbor from the static routes) and then deconfiguring the static BFD neighbor.

BFD Support for OSPFv3

Bidirectional Forwarding Detection (BFD) supports OSPFv3.

How to Configure Bidirectional Forwarding Detection for IPv6

Specifying a Static BFDv6 Neighbor

An IPv6 static BFDv6 neighbor is specified separately from an IPv6 static route. An IPv6 static BFDv6 neighbor must be fully configured with the interface and neighbor address and must be directly attached to the local router.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ipv6 route static bfd** [**vrf** *vrf-name*] *interface-type interface-number ipv6-address* [**unassociated**]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**ipv6 route static bfd** [**vrf** *vrf-name*] *interface-type interface-number ipv6-address* [**unassociated**] **Example:** `Device(config)# ipv6 route static bfd gigabitethernet 0/0/0 2001::1`	Specifies static route IPv6 BFDv6 neighbors.

Associating an IPv6 Static Route with a BFDv6 Neighbor

IPv6 static routes are automatically associated with a static BFDv6 neighbor. A static neighbor is associated with a BFDv6 neighbor if the static next-hop explicitly matches the BFDv6 neighbor.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ipv6 route static bfd** [**vrf** *vrf-name*] *interface-type interface-number ipv6-address* [**unassociated**]
4. **ipv6 route** [**vrf** *vrf-name*] *ipv6-prefix / prefix-length* {*ipv6-address* | *interface-type interface-number ipv6-address*]} [**nexthop-vrf** [*vrf-name1* | **default**]] [*administrative-distance*] [*administrative-multicast-distance* | **unicast** | **multicast**] [*next-hop-address*] [**tag** *tag*]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**ipv6 route static bfd** [**vrf** *vrf-name*] *interface-type interface-number ipv6-address* [**unassociated**] **Example:** `Device(config)# ipv6 route static bfd gigabitethernet 0/0/0 2001::1`	Specifies static route BFDv6 neighbors.
Step 4	**ipv6 route** [**vrf** *vrf-name*] *ipv6-prefix / prefix-length* {*ipv6-address* | *interface-type interface-number ipv6-address*]} [**nexthop-vrf** [*vrf-name1* | **default**]] [*administrative-distance*] [*administrative-multicast-distance* | **unicast** | **multicast**] [*next-hop-address*] [**tag** *tag*] **Example:** `Device(config)# ipv6 route 2001:DB8::/64 gigabitethernet 0/0/0 2001::1`	Establishes static IPv6 routes.

Configuring BFD Support for OSPFv3

This section describes the procedures for configuring BFD support for OSPFv3, so that OSPFv3 is a registered protocol with BFD and will receive forwarding path detection failure messages from BFD. You can either configure BFD support for OSPFv3 globally on all interfaces or configure it selectively on one or more interfaces.

There are two methods for enabling BFD support for OSPFv3:

- You can enable BFD for all of the interfaces for which OSPFv3 is routing by using the **bfd all-interfaces** command in router configuration mode. You can disable BFD support on individual interfaces using the **ipv6 ospf bfd disable** command in interface configuration mode.
- You can enable BFD for a subset of the interfaces for which OSPFv3 is routing by using the **ipv6 ospf bfd** command in interface configuration mode.

Note OSPF will only initiate BFD sessions for OSPF neighbors that are in the FULL state.

Configuring Baseline BFD Session Parameters on the Interface

Repeat this task for each interface over which you want to run BFD sessions to BFD neighbors.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type number*
4. **bfd interval** *milliseconds* **min_rx** *milliseconds* **multiplier** *interval-multiplier*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**interface** *type number* **Example:** `Router(config)# interface gigabitethernet 0/0/0`	Specifies an interface type and number, and places the router in interface configuration mode.
Step 4	**bfd interval** *milliseconds* **min_rx** *milliseconds* **multiplier** *interval-multiplier* **Example:** `Router(config-if)# bfd interval 50 min_rx 50 multiplier 5`	Enables BFD on the interface.

Configuring BFD Support for OSPFv3 for All Interfaces

OSPFv3 must be running on all participating routers. The baseline parameters for BFD sessions on the interfaces over which you want to run BFD sessions to BFD neighbors must be configured.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ipv6 router ospf** *process-id* [**vrf** *vpn-name*]
4. **bfd all-interfaces**
5. **exit**
6. **show bfd neighbors** [**vrf** *vrf-name*] [**client** {**bgp** | **eigrp** | **isis** | **ospf** | **rsvp** | **te-frr**}] [*ip-address* | **ipv6** *ipv6-address*] [**details**]
7. **show ipv6 ospf** [*process-id*] [*area-id*] [**rate-limit**]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**ipv6 router ospf** *process-id* [**vrf** *vpn-name*] **Example:** `Device(config)# ipv6 router ospf 2`	Configures an OSPFv3 routing process.
Step 4	**bfd all-interfaces** **Example:** `Device(config-router)# bfd all-interfaces`	Enables BFD for all interfaces participating in the routing process.

	Command or Action	Purpose
Step 5	**exit** **Example:** `Device(config-router)# exit`	Enter this command twice to go to privileged EXEC mode.
Step 6	**show bfd neighbors** [**vrf** *vrf-name*] [**client** {**bgp** | **eigrp** | **isis** | **ospf** | **rsvp** | **te-frr**}] [*ip-address* | **ipv6** *ipv6-address*] [**details**] **Example:** `Device# show bfd neighbors detail`	(Optional) Displays a line-by-line listing of existing BFD adjacencies.
Step 7	**show ipv6 ospf** [*process-id*] [*area-id*] [**rate-limit**] **Example:** `Device# show ipv6 ospf`	(Optional) Displays general information about OSPFv3 routing processes.

Configuring BFDv6 Support for OSPFv3 on One or More OPSFv3 Interfaces

OSPFv3 must be running on all participating routers. The baseline parameters for BFD sessions on the interfaces over which you want to run BFD sessions to BFD neighbors must be configured. See the Configuring Baseline BFD Session Parameters on the Interface, page 105 section for more information.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type number*
4. **ipv6 ospf bfd** [**disable**]
5. **exit**
6. **show bfd neighbors** [**vrf** *vrf-name*] [**client** {**bgp** | **eigrp** | **isis** | **ospf** | **rsvp** | **te-frr**}] [*ip-address*| **ipv6** *ipv6-address*] [**details**]
7. **show ipv6 ospf** [*process-id*] [*area-id*] [**rate-limit**]

DETAILED STEPS

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

	Command or Action	Purpose
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**interface** *type number* **Example:** `Router(config)# interface gigabitethernet 0/0/0`	Specifies an interface type and number, and places the router in interface configuration mode.
Step 4	**ipv6 ospf bfd** [**disable**] **Example:** `Router(config-if)# ipv6 ospf bfd`	Enables BFD on a per-interface basis for one or more interfaces associated with the OSPFv3 routing process.
Step 5	**exit** **Example:** `Router(config-router)# exit`	Enter this command twice to go to privileged EXEC mode.
Step 6	**show bfd neighbors** [**vrf** *vrf-name*] [**client** {**bgp** | **eigrp** | **isis** | **ospf** | **rsvp** | **te-frr**}] [*ip-address*| **ipv6** *ipv6-address*] [**details**] **Example:** `Router# show bfd neighbors detail`	(Optional) Displays a line-by-line listing of existing BFD adjacencies.
Step 7	**show ipv6 ospf** [*process-id*] [*area-id*] [**rate-limit**] **Example:** `Router# show ipv6 ospf`	(Optional) Displays general information about OSPFv3 routing processes.

Retrieving BFDv6 Information for Monitoring and Troubleshooting

SUMMARY STEPS

1. **enable**
2. **monitor event ipv6 static** [**enable** | **disable**]
3. **show ipv6 static** [*ipv6-address* | *ipv6-prefix/prefix-length*] [**interface** *type number* | **recursive**] [**vrf** *vrf-name*] [**bfd**] [**detail**
4. **show ipv6 static** [*ipv6-address* | *ipv6-prefix/prefix-length*] [**interface** *type number* | **recursive**] [**vrf** *vrf-name*] [**bfd**] [**detail**]
5. **debug ipv6 static**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**monitor event ipv6 static** [**enable**	**disable**] **Example:** `Device# monitor event ipv6 static enable`	Enables the use of event trace to monitor the operation of the IPv6 static and IPv6 static BFDv6 neighbors.
Step 3	**show ipv6 static** [*ipv6-address*	*ipv6-prefix/prefix-length*] [**interface** *type number*	**recursive**] [**vrf** *vrf-name*] [**bfd**] [**detail** **Example:** `Device# show ipv6 static vrf vrf1 detail`	Displays the BFDv6 status for a static route associated with a static BFDv6 neighbor.
Step 4	**show ipv6 static** [*ipv6-address*	*ipv6-prefix/prefix-length*] [**interface** *type number*	**recursive**] [**vrf** *vrf-name*] [**bfd**] [**detail**] **Example:** `Device# show ipv6 static vrf vrf1 bfd`	Displays static BFDv6 neighbors and associated static routes.
Step 5	**debug ipv6 static** **Example:** `Device# debug ipv6 static`	Enables BFDv6 debugging.

Configuration Examples for Bidirectional Forwarding Detection for IPv6

Example: Specifying an IPv6 Static BFDv6 Neighbor

The following example specifies a fully configured IPv6 static BFDv6 neighbor. The interface is GigabitEthernet 0/0/0 and the neighbor address is 2001::1.

```
Device(config)# ipv6 route static bfd gigabitethernet 0/0/0 2001::1
```

Example: Associating an IPv6 Static Route with a BFDv6 Neighbor

In this example, the IPv6 static route 2001:DB8::/32 is associated with the BFDv6 neighbor 2001::1 over the GigabitEthernet 0/0/0 interface:

```
Device(config)# ipv6 route static bfd gigabitethernet 0/0/0 2001::1
Device(config)# ipv6 route 2001:DB8::/32 gigabitethernet 0/0/0 2001::1
```

Additional References

Related Documents

Related Topic	Document Title
OSPF for IPv6	"Implementing OSPF for IPv6," *Cisco IOS IPv6 Configuration Guide*
IPv6 static routes	"Implementing Static Routes for IPv6," *Cisco IOS IPv6 Configuration Guide*

Standards

Standard	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
None	To locate and download MIBs for selected platforms, Cisco software releases, and feature sets, use Cisco MIB Locator found at the following URL: http://www.cisco.com/go/mibs

RFCs

RFC	Title
draft-ietf-bfd-v4v6-1hop-07.txt	*BFD for IPv4 and IPv6 (Single Hop)*

Technical Assistance

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

Feature Information for Implementing Bidirectional Forwarding for IPv6

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Table 8 Feature Information for Implementing Bidirectional Forwarding for IPv6

Feature Name	Releases	Feature Information
IPv6 Routing: Static Route Support for BFD over IPv6	Cisco IOS XE Release 2.1	BFD for IPv6 is used to verify next-hop reachability for IPv6 static routes. The following commands were introduced or modified by this feature: **debug ipv6 static**, **ipv6 route**, **ipv6 route static bfd**, **monitor event ipv6 static**, **show ipv6 static**
OSPFv3 for BFD	Cisco IOS XE Release 2.1	BFD supports the dynamic routing protocol OSPF for IPv6 (OSPFv3). The following commands were introduced or modified by this feature: **bfd all-interfaces**, **bfd interval**, **ipv6 ospf bfd**, **ipv6 router ospf**, **show bfd neighbors**

Implementing DHCP for IPv6

This module describes how to configure Dynamic Host Configuration Protocol (DHCP) for IPv6.

Finding Feature Information

Your software release may not support all the features documented in this module. For the latest caveats and feature information, see Bug Search Tool and the release notes for your platform and software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the feature information table at the end of this module.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Information About Implementing DHCP for IPv6

DHCPv6 Prefix Delegation

The IPv6 Access Services—DHCPv6 Prefix Delegation feature can be used to manage link, subnet, and site addressing changes. Dynamic Host Configuration Protocol for IPv6 (DHCPv6) can be used in environments to deliver stateful and stateless information. The definitions are given below:

- Stateful—Address assignment is centrally managed and clients must obtain configuration information that is not available through protocols such as address autoconfiguration and neighbor discovery.
- Stateless—Stateless configuration parameters do not require a server to maintain any dynamic state for individual clients, such as Domain Name System (DNS) server addresses and domain search list options.

Extensions to DHCPv6 also enable prefix delegation, through which an ISP can automate the process of assigning prefixes to a customer for use within the customer's network. Prefix delegation occurs between a

provider edge (PE) device and customer premises equipment (CPE) using the DHCPv6 prefix delegation option. Once the ISP has delegated prefixes to a customer, the customer may further subnet and assign prefixes to the links in the customer's network.

Configuring Nodes Without Prefix Delegation

Stateless DHCPv6 allows DHCPv6 to be used for configuring a node with parameters that do not require a server to maintain any dynamic state for the node. The use of stateless DHCP is controlled by router advertisement (RA) messages multicasted by routers. The Cisco IOS XE DHCPv6 client will invoke stateless DHCPv6 when it receives an RA. The Cisco IOS XE DHCPv6 server will respond to a stateless DHCPv6 request with configuration parameters, such as the DNS servers and domain search list options.

Client and Server Identification

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

When a DHCPv6 client requests two prefixes with the same DUID but with different IAIDs on two different interfaces, these prefixes are considered to be for two different clients, and the interface information is maintained for both.

Rapid Commit

The DHCPv6 client can obtain configuration parameters from a server either through a rapid two-message exchange (solicit, reply) or through a normal four-message exchange (solicit, advertise, request, reply). By default, the four-message exchange is used. When the rapid-commit option is enabled by both client and server, the two-message exchange is used.

DHCPv6 Client, Server, and Relay Functions

The DHCPv6 client, server, and relay functions are mutually exclusive on an interface. When one of these functions is enabled and a user tries to configure a different function on the same interface, one of the following messages is displayed: "Interface is in DHCP client mode," "Interface is in DHCP server mode," or "Interface is in DHCP relay mode."

The following sections describe these functions:

Client Function

The DHCPv6 client function can be enabled on individual IPv6-enabled interfaces.

The DHCPv6 client can request and accept those configuration parameters that do not require a server to maintain any dynamic state for individual clients, such as DNS server addresses and domain search list options.

The DHCPv6 client can also request the delegation of prefixes. The prefixes acquired from a delegating router will be stored in a local IPv6 general prefix pool. The prefixes in the general prefix pool can then be referred to from other applications; for example, the general prefix pool can be used to number router downstream interfaces.

Server Selection

A DHCPv6 client builds a list of potential servers by sending a solicit message and by collecting advertise message replies from servers. These messages are ranked based on the preference value, and servers may add a preference option to their advertise messages explicitly stating their preference value. If the client needs to acquire prefixes from servers, only servers that have advertised prefixes are considered.

IAPD and IAID

An Identity Association for Prefix Delegation (IAPD) is a collection of prefixes assigned to a requesting router. A requesting router may have more than one IAPD; for example, one for each of its interfaces.

Each IAPD is identified by an IAID. The IAID is chosen by the requesting router and is unique among the IAPD IAIDs on the requesting router. IAIDs are made consistent across reboots by using information from the associated network interface, which is assumed to be permanently attached to the device.

Server Function

The DHCPv6 server function can be enabled on individual IPv6-enabled interfaces.

The DHCPv6 server can provide configuration parameters that do not require the server to maintain any dynamic state for individual clients, such as DNS server addresses and domain search list options. The DHCPv6 server may be configured to perform prefix delegation.

All the configuration parameters for clients are independently configured into DHCPv6 configuration pools, which are stored in the NVRAM. A configuration pool can be associated with a particular DHCPv6 server on an interface when it is started. Prefixes that are to be delegated to clients may be specified either as a list of preassigned prefixes for a particular client or as IPv6 local prefix pools that are also stored in the NVRAM. The list of manually configured prefixes or IPv6 local prefix pools can be referenced and used by DHCPv6 configuration pools.

The DHCPv6 server maintains an automatic binding table in memory to track the assignment of some configuration parameters, such as prefixes between the server and its clients. Automatic bindings can be stored permanently in the database agent, such as a remote TFTP server or a local NVRAM file system.

Configuration Information Pool

A DHCPv6 configuration information pool is a named entity that includes information about available configuration parameters and policies that the control assignment of the parameters to clients from the pool. A pool is configured independently and is associated with the DHCPv6 service through the CLI.

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

- Prefix delegation information, which includes:
  - A prefix pool name and associated preferred and valid lifetimes
  - A list of available prefixes for a particular client and associated preferred and valid lifetimes

- A list of IPv6 addresses of DNS servers
- A domain search list, which is a string containing domain names for the DNS resolution

Prefix Assignment

A prefix-delegating router (DHCPv6 server) selects prefixes to be assigned to a requesting router (DHCPv6 client) upon receiving a request from the client. The server can select prefixes for a requesting client by using static and dynamic assignment mechanisms. Administrators can manually configure a list of prefixes and associated preferred and valid lifetimes for an IAPD of a specific client that is identified by its DUID.

When the delegating router receives a request from a client, it checks if there is a static binding configured for the IAPD in the client's message. If a static binding is present, the prefixes in the binding are returned to the client. If no such binding is found, the server attempts to assign prefixes for the client from other sources.

The Cisco IOS XE DHCPv6 server can assign prefixes dynamically from an IPv6 local prefix pool. When the server receives a prefix request from a client, it attempts to obtain unassigned prefixes from the pool. After the client releases the previously assigned prefixes, the server returns them to the pool for reassignment.

An IPv6 prefix delegating router can also select prefixes for a requesting router based on an external authority such as a RADIUS server using the Framed-IPv6-Prefix attribute.

Automatic Binding

Each DHCPv6 configuration pool has an associated binding table. The binding table contains records of all prefixes in the configuration pool that have been explicitly delegated to clients. Each entry in the binding table contains the following information:

- Client DUID.
- Client IPv6 address.
- A list of IAPDs associated with the client.
- A list of prefixes delegated to each IAPD.
- Preferred and valid lifetimes for each prefix.
- The configuration pool to which this binding table belongs.
- The network interface on which the server that is using the pool is running.

A binding table entry is automatically created whenever a prefix is delegated to a client from the configuration pool, and the entry is updated when the client renews, rebinds, or confirms the prefix delegation. A binding table entry is deleted when the client voluntarily releases all the prefixes in the binding, the valid lifetimes of all prefixes have expired, or administrators run the **clear ipv6 dhcp binding** command.

Binding Database

Each permanent storage to which the binding database is saved is called the database agent. A database agent can be a remote host, such as an FTP server, or a local file system, such as the NVRAM.

Automatic bindings are maintained in the RAM and can be saved to some permanent storage so that information about configurations, such as prefixes assigned to clients, is not lost after a system reload. The bindings are stored as text records for easy maintenance. Each record contains the following information:

- DHCPv6 pool name from which the configuration was assigned to the client.
- Interface identifier from which the client requests were received.
- The client IPv6 address.
- The client DUID.

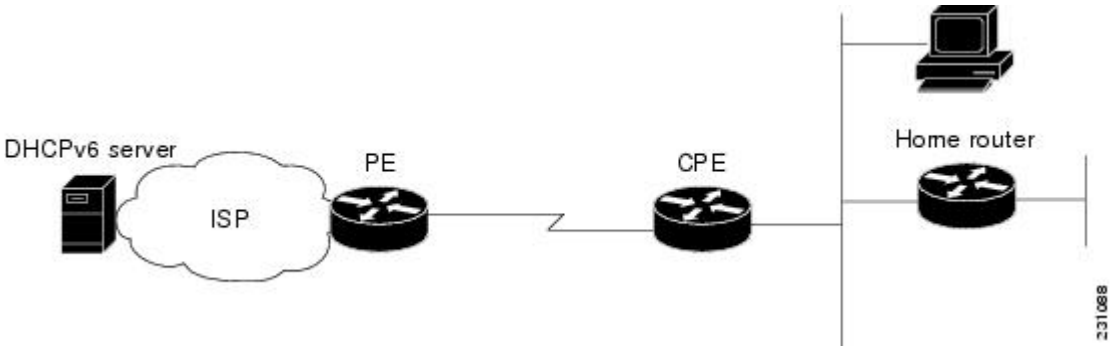
- IAID of the IAPD.
- Prefix delegated to the client.
- The prefix length.
- The prefix preferred lifetime in seconds.
- The prefix valid lifetime in seconds.
- The prefix expiration time stamp.
- Optional local prefix pool name from which the prefix was assigned.

DHCPv6 Server Stateless Autoconfiguration

Hierarchical DHCPv6 for stateless configuration parameters allows a stateless or stateful DHCPv6 client to export configuration parameters (DHCPv6 options) to a local DHCPv6 server pool. The local DHCPv6 server can then provide the imported configuration parameters to other DHCPv6 clients.

The figure below shows a typical broadband deployment.

Figure 20 Broadband Topology



The CPE interface towards the PE can be a stateless or stateful DHCPv6 client. In either case, the ISP-side DHCPv6 server may provide configuration parameters such as DNS server addresses, domain names, and Simple Network Time Protocol (SNTP) servers to the DHCP client on the CPE. Such information can be specific to ISPs.

In addition to being a DHCPv6 client (for example, towards the ISP), the CPE may act as a DHCPv6 server to the home network. For example, neighbor discovery followed by a stateless or stateful DHCPv6 client can occur on the link between the CPE and the home devices (such as the home router or PC). In some cases, the information to be provided to the home network is the same as that obtained from the ISP-side DHCPv6 server. Because this information can be dynamically changed, it cannot be hard-configured in the CPE's configuration. Therefore, the DHCPv6 component on the CPE allows automatic importing of configuration parameters from the DHCPv6 client to the DHCPv6 server pool.

DHCPv6 supports the following options for IPv6 on the server:

Information Refresh Server Option

The DHCPv6 information refresh option can specify a maximum limit for the length of time a client should wait before refreshing the information retrieved from DHCPv6. This option is used with stateless DHCPv6 because there are no addresses or other entities with lifetimes that can tell the client when to contact the DHCPv6 server to refresh its configuration.

NIS- and NIS+-Related Server Options

Users can configure the network information service (NIS) or NIS plus (NIS+) address or domain name of a DHCPv6 server using NIS- and NIS+-related options, and then import that information to the DHCPv6 client.

SIP Server Options

Session Initiation Protocol (SIP) server options contain either a list of domain names or a list of IPv6 addresses that can be mapped to one or more SIP outbound proxy servers. One option carries a list of domain names, and the other option carries a list of 128-bit IPv6 addresses.

SIP is an application-layer control protocol that can establish, modify, and terminate multimedia sessions or calls. A SIP system has several logical components: user agents, proxy servers, redirect servers, and registrars. User agents may contain SIP clients; proxy servers always contain SIP clients.

SNTP Server Option

The SNTP server option provides a list of one or more IPv6 addresses of SNTP servers available to the client for synchronization. Clients use these SNTP servers to synchronize their system time to that of the standard time servers. The DHCPv6 server may list the SNTP servers in decreasing order of preference, but clients treat the list of SNTP servers as an ordered list.

DHCP Relay Agent

A DHCP relay agent, which may reside on the client's link, is used to relay messages between the client and server. DHCP relay agent operation is transparent to the client. A client locates a DHCP server using a reserved, link-scoped multicast address. Therefore, it is a requirement for direct communication between the client and the server that the client and the server be attached to the same link. However, in some situations in which ease of management, economy, or scalability is a concern, it is desirable to allow a DHCP client to send a message to a DHCP server that is not connected to the same link.

DHCPv6 Relay Agent Notification for Prefix Delegation

DHCPv6 relay agent notification for prefix delegation allows the router working as a DHCPv6 relay agent to find prefix delegation options by reviewing the contents of a DHCPv6 RELAY-REPLY packet that is being relayed by the relay agent to the client. When a prefix delegation option is found by the relay agent, the relay agent extracts the information about the prefix being delegated and inserts an IPv6 static route matching the prefix delegation information onto the relay agent. Future packets destined to that prefix via relay will be forwarded based on the information contained in the prefix delegation. The IPv6 static route is then left in the routing table until the prefix delegation lease time expires or the relay agent receives a release packet from the client releasing the prefix delegation.

No user configuration is required for this feature. Static route management is done automatically by the relay agent.

The IPv6 routes are added when the relay agent relays a RELAY-REPLY packet, and the IPv6 routes are deleted when the prefix delegation lease time expires or the relay agent receives a release message. An IPv6 static route in the routing table of the relay agent can be updated when the prefix delegation lease time is extended.

This feature leaves a static IPv6 route on the routing table of the relay agent. This registered IPv6 address allows unicast reverse packet forwarding (uRPF) to work by allowing the router doing the reverse lookup to confirm that the IPv6 address on the relay agent is not malformed or spoofed. The static route left in the routing table of the relay agent can be redistributed to other routing protocols to advertise the subnets to other nodes. The static routes will be removed when an DHCP_DECLINE message is sent by the client.

DHCPv6 Relay Options: Remote-ID for GigabitEthernet and FastEthernet Interfaces

This feature adds the remote identification (remote-ID) option to relayed (RELAY-FORWARD) DHCPv6 packets.

The remote-ID option provides information to the DHCPv6 server, including port information, the system's DUID, and the VLAN ID. Collectively, this information can be used to uniquely identify both the relay and the port on the relay through which the client's packet arrived. The DHCPv6 server uses this information to select parameters specific to a particular user, host, or subscriber modem.

This feature introduces no user configuration. Because the addition of the remote-ID option to the RELAY-FORWARD packet occurs automatically, no user configuration is necessary.

The DHCPv6 server does not need to echo the remote-ID option in the RELAY-REPLY packet. Internet Assigned Numbers Authority (IANA) has assigned the DHCPv6 option code 37 for the relay agent remote-ID option.

If the remote-ID option is included in the RELAY-REPLY packet, the option is stripped out of the packet before the packet is relayed to the client.

DHCPv6 Relay Options: Reload Persistent Interface-ID

This feature makes the interface-ID option, which is used by relay agents to decide which interface should be used when forwarding a RELAY-REPLY packet, persistent. A persistent interface-ID option will not change if the router acting as a relay agent goes offline (such as during a reload or a power outage). When the router acting as a relay agent returns online, it is possible that changes to the internal interface index of the relay agent may have occurred in certain scenarios (such as cases where the relay agent reboots and has a change in the number of interfaces in the interface index, or the relay agents boots up and has more virtual interfaces than it did before the reboot). This feature prevents this scenario from causing any problems.

This feature changes the DHCPv6 interface-ID option to be expressed as simply the short form of the interface name. This syntax helps avoid potential problems that could arise due to physical or logical interfaces changing on the relay agent after a reload.

DHCPv6 Relay Chaining

DHCPv6 messages can be relayed through multiple relay agents. This configuration is called *relay chaining*. A relay chaining configuration can be supported only when each relay agent adds certain information to DHCPv6 messages before relaying them. The additional information helps in relaying the DHCPv6 reply back to the DHCPv6 client through the same path.

The delegated IPv6 prefix must be routable in order to be useful. The actual DHCPv6 Prefix Delegation (PD) client may not be permitted to inject routes into the delegating network. In service-provider (SP) networks, for example, an edge router typically acts as a DHCPv6 relay agent, and this edge router often has the responsibility to maintain routes within the SP network for clients' PD bindings. In the event that DHCPv6 requests and responses are relayed through a chain of DHCPv6 relays, there may be a need to introduce appropriate routes (particularly with DHCPv6 PD) in the Forwarding Information Base (FIB) so that routing is handled transparently.

DHCPv6 Server and Relay—MPLS VPN Support

To facilitate managed central services in a Multiprotocol Label Switching (MPLS)-based network, DHCPv6 must be made MPLS-aware so a single resource can be used to serve multiple virtual private networks (VPNs) instead of dedicating a resource to a single VPN.

The DHCPv6 server implementation of MPLS VPN support allows a per-pool configuration so DHCPv6 pools can be associated with a VPN routing and forwarding (VRF) instance. The DHCPv6 server

differentiates clients from various VRFs and assigns an IPv6 prefix accordingly from the respective VRF pools. Meanwhile, the DHCPv6 bindings store clients' VRF information.

The DHCPv6 relay implementation allows the configuration of the destination VRF instance to which the relay messages will be forwarded. The relay adds the client's VPN information while forwarding the client's DHCPv6 requests toward the server, and the relay then processes the client's VPN information in reply packets from server.

The relay adds IPv6 static routes for delegated prefixes in corresponding clients' VRF, and the relay's high availability (HA) functionality synchronizes the VRF information while synchronizing static routes created by the relay process.

The DHCPv6 relay and server VRF-aware features are disabled by default for backward compatibility.

How to Implement DHCP for IPv6

Configuring the DHCPv6 Server Function

The tasks in the following sections explain how to configure DHCPv6 server function:

Creating and Configuring the DHCPv6 Configuration Pool

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ipv6 dhcp pool** *poolname*
4. **domain-name** *domain*
5. **dns-server** *ipv6-address*
6. **prefix-delegation** *ipv6-prefix* / *prefix-length client-DUID* [**iaid** *iaid*] [*lifetime*]
7. **prefix-delegation pool** *poolname* [**lifetime** {*valid-lifetime* | *preferred-lifetime*}]
8. **exit**
9. **interface** *type number*
10. **ipv6 dhcp server** *poolname* [**rapid-commit**] [**preference** *value*] [**allow-hint**]
11. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**ipv6 dhcp pool** *poolname* **Example:** `Router(config)# ipv6 dhcp pool pool1`	Configures a DHCPv6 configuration information pool and enters DHCPv6 pool configuration mode.
Step 4	**domain-name** *domain* **Example:** `Router(config-dhcp)# domain-name example.com`	Configures a domain name for a DHCPv6 client.

	Command or Action	Purpose
Step 5	**dns-server** *ipv6-address* **Example:** `Router(config-dhcp)# dns-server 2001:DB8:3000:3000::42`	Specifies the DNS IPv6 servers available to a DHCPv6 client.
Step 6	**prefix-delegation** *ipv6-prefix* / *prefix-length client-DUID* [**iaid** *iaid*] [*lifetime*] **Example:** `Router(config-dhcp)# prefix-delegation 2001:DB8:1263::/48 0005000400F1A4D070D03`	Specifies a manually configured numeric prefix to be delegated to a specified client's IAPD.
Step 7	**prefix-delegation pool** *poolname* [**lifetime** {*valid-lifetime* | *preferred-lifetime*}] **Example:** `Router(config-dhcp)# prefix-delegation pool pool1 lifetime 1800 60`	Specifies a named IPv6 local prefix pool from which prefixes are delegated to DHCPv6 clients.
Step 8	**exit** **Example:** `Router(config-dhcp)# exit`	Exits DHCPv6 pool configuration mode configuration mode, and returns the router to global configuration mode.
Step 9	**interface** *type number* **Example:** `Router(config)# interface serial 3`	Specifies an interface type and number, and places the router in interface configuration mode.
Step 10	**ipv6 dhcp server** *poolname* [**rapid-commit**] [**preference** *value*] [**allow-hint**] **Example:** `Router(config-if)# ipv6 dhcp server pool1`	Enables DHCPv6 on an interface.
Step 11	**end** **Example:** `Router(config-if)# end`	Returns to privileged EXEC mode.

Configuring a Binding Database Agent for the Server Function

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ipv6 dhcp database** *agent* [**write-delay** *seconds*] [**timeout** *seconds*]
4. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**ipv6 dhcp database** *agent* [**write-delay** *seconds*] [**timeout** *seconds*] **Example:** `Router(config)# ipv6 dhcp database tftp://10.0.0.1/dhcp-binding`	Specifies DHCPv6 binding database agent parameters.
Step 4	**end** **Example:** `Router(config)# end`	Returns to privileged EXEC mode.

Configuring the DHCPv6 Client Function

General prefixes can be defined dynamically from a prefix received by a DHCPv6 prefix delegation client. The delegated prefix is stored in a general prefix.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type number*
4. **ipv6 dhcp client pd** {*prefix-name* | **hint** *ipv6-prefix*} [**rapid-commit**]
5. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**interface** *type number* **Example:** `Router(config)# interface fastethernet 0/0/0`	Specifies an interface type and number, and enters interface configuration mode.
Step 4	**ipv6 dhcp client pd** {*prefix-name* | **hint** *ipv6-prefix*} [**rapid-commit**] **Example:** `Router(config-if)# ipv6 dhcp client pd dhcp-prefix`	Enables the DHCPv6 client process and enables a request for prefix delegation through a specified interface.
Step 5	**end** **Example:** `Router(config-if)# end`	Returns to privileged EXEC mode.

Configuring the DHCPv6 Relay Agent

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type number*
4. **ipv6 dhcp relay destination** *ipv6-address* [*interface-type interface-number*]
5. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**interface** *type number* **Example:** `Router(config)# interface gigabitethernet 4/2/0`	Specifies an interface type and number, and enters interface configuration mode.
Step 4	**ipv6 dhcp relay destination** *ipv6-address* [*interface-type interface-number*] **Example:** `Router(config-if) ipv6 dhcp relay destination FE80::250:A2FF:FEBF:A056 gigabitethernet 4/3/0`	Specifies a destination address to which client packets are forwarded and enables the DHCPv6 relay service on the interface.
Step 5	**end** **Example:** `Router(config-if)# end`	Returns to privileged EXEC mode.

Configuring Route Addition for Relay and Server

To enable route addition by DHCPv6 relay and server for the delegated prefix, use the **ipv6 dhcp iapd-route-add** command in global configuration mode.

To add routes for individually assigned IPv6 addresses on the relay or server, use the **ipv6 dhcp iana-route-add** command in global configuration mode

Configuring the Stateless DHCPv6 Function

The server maintains no state related to clients; for example, no prefix pools and records of allocation are maintained. Therefore, this function is “stateless” DHCPv6.

Configuring the Stateless DHCPv6 Server

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ipv6 dhcp pool** *poolname*
4. **dns-server** *ipv6-address*
5. **domain-name** *domain*
6. **exit**
7. **interface** *type number*
8. **ipv6 dhcp server** *poolname* [**rapid-commit**] [**preference** *value*] [**allow-hint**]
9. **ipv6 nd other-config-flag**
10. **end**

DETAILED STEPS

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

	Command or Action	Purpose
Step 3	**ipv6 dhcp pool** *poolname* **Example:** `Router(config)# ipv6 dhcp pool dhcp-pool`	Configures a DHCPv6 configuration information pool and enters DHCPv6 pool configuration mode.
Step 4	**dns-server** *ipv6-address* **Example:** `Router(config-dhcp) dns-server 2001:DB8:3000:3000::42`	Specifies the DNS IPv6 servers available to a DHCPv6 client.
Step 5	**domain-name** *domain* **Example:** `Router(config-dhcp)# domain-name domain1.com`	Configures a domain name for a DHCPv6 client.
Step 6	**exit** **Example:** `Router(config-dhcp)# exit`	Exits DHCPv6 pool configuration mode, and returns the router to global configuration mode.
Step 7	**interface** *type number* **Example:** `Router(config)# interface serial 3`	Specifies an interface type and number, and places the router in interface configuration mode.
Step 8	**ipv6 dhcp server** *poolname* [**rapid-commit**] [**preference** *value*] [**allow-hint**] **Example:** `Router(config-if)# ipv6 dhcp server dhcp-pool`	Enables DHCPv6 on an interface.
Step 9	**ipv6 nd other-config-flag** **Example:** `Router(config-if)# ipv6 nd other-config-flag`	Sets the “other stateful configuration” flag in IPv6 RAs.

	Command or Action	Purpose
Step 10	**end** **Example:** `Router(config-if)# end`	Returns to privileged EXEC mode.

Enabling Processing of Packets with Source Routing Header Options

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ipv6 source-route**
4. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**ipv6 source-route** **Example:** `Router(config)# ipv6 source-route`	Enables the processing of the IPv6 type 0 routing header.
Step 4	**end** **Example:** `Router(config)# end`	Returns to privileged EXEC mode.

Configuring the DHCPv6 Server Options

Configuring the Information Refresh Server Option

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ipv6 dhcp pool** *poolname*
4. **information refresh** {*days* [*hours minutes*] | **infinity**}
5. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**ipv6 dhcp pool** *poolname* **Example:** `Router(config)# ipv6 dhcp pool pool1`	Configures a DHCPv6 configuration information pool and enters DHCPv6 pool configuration mode.

	Command or Action	Purpose
Step 4	**information refresh** {*days* [*hours minutes*] | **infinity**} **Example:** `Router(config-dhcp)# information refresh 1 1 1`	Specifies the information refresh time to be sent to the client.
Step 5	**end** **Example:** `Router(config-dhcp)# end`	Returns to privileged EXEC mode.

Importing the Information Refresh Server Option

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ipv6 dhcp pool** *poolname*
4. **import information refresh**
5. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**ipv6 dhcp pool** *poolname* **Example:** `Router(config)# ipv6 dhcp pool pool1`	Configures a DHCPv6 configuration information pool and enters DHCPv6 pool configuration mode.

	Command or Action	Purpose
Step 4	**import information refresh** **Example:** `Router(config-dhcp)# import information refresh`	Imports the information refresh time option to a DHCPv6 client.
Step 5	**end** **Example:** `Router(config-dhcp)# end`	Returns to privileged EXEC mode.

Configuring NIS- and NISP-Related Server Options

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ipv6 dhcp pool** *poolname*
4. **nis address** *ipv6-address*
5. **nis domain-name** *domain-name*
6. **nisp address** *ipv6-address*
7. **nisp domain-name** *domain-name*
8. **end**

DETAILED STEPS

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

Command or Action	Purpose
Step 3 **ipv6 dhcp pool** *poolname* **Example:** `Router(config)# ipv6 dhcp pool pool1`	Configures a DHCPv6 configuration information pool and enters DHCPv6 pool configuration mode.
Step 4 **nis address** *ipv6-address* **Example:** `Router(config-dhcp)# nis address 2001:DB8:1000:1000::30`	Specifies the NIS address of an IPv6 server to be sent to the client.
Step 5 **nis domain-name** *domain-name* **Example:** `Router(config-dhcp)# nis domain-name domain1`	Enables a server to convey a client's NIS domain name information to the client.
Step 6 **nisp address** *ipv6-address* **Example:** `Router(config-dhcp)# nisp address 2001:DB8:3000:3000::42`	Specifies the NIS+ address of an IPv6 server to be sent to the DHCPv6 client.
Step 7 **nisp domain-name** *domain-name* **Example:** `Router(config-dhcp)# nisp domain-name domain2`	Enables a server to convey a client's NIS+ domain name information to the DHCPv6 client.
Step 8 **end** **Example:** `Router(config-dhcp)# end`	Returns to privileged EXEC mode.

Importing NIS- and NIS+-Related Server Options

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ipv6 dhcp pool** *poolname*
4. **import nis address**
5. **import nis domain-name**
6. **import nisp address**
7. **import nisp domain-name**
8. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**ipv6 dhcp pool** *poolname* **Example:** `Router(config)# ipv6 dhcp pool pool1`	Configures a DHCPv6 configuration information pool and enters DHCPv6 pool configuration mode.
Step 4	**import nis address** **Example:** `Router(config-dhcp)# import nis address`	Imports the NIS servers option to a DHCPv6 client.
Step 5	**import nis domain-name** **Example:** `Router(config-dhcp)# import nis domain-name`	Imports the NIS domain name option to a DHCPv6 client.

	Command or Action	Purpose
Step 6	**import nisp address** **Example:** `Router(config-dhcp)# import nisp address`	Imports the NISP address option to a DHCPv6 client.
Step 7	**import nisp domain-name** **Example:** `Router(config-dhcp)# import nisp domain-name`	Imports the NISP domain name option to a DHCPv6 client.
Step 8	**end** **Example:** `Router(config-dhcp)# end`	Returns to privileged EXEC mode.

Importing SIP Server Options

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ipv6 dhcp pool** *poolname*
4. **import sip address**
5. **import sip domain-name**
6. **end**

DETAILED STEPS

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

	Command or Action	Purpose
Step 3	**ipv6 dhcp pool** *poolname* **Example:** `Router(config)# ipv6 dhcp pool pool1`	Configures a DHCPv6 configuration information pool and enters DHCPv6 pool configuration mode.
Step 4	**import sip address** **Example:** `Router(config-dhcp)# import sip address`	Imports the SIP server IPv6 address list option to the outbound SIP proxy server.
Step 5	**import sip domain-name** **Example:** `Router(config-dhcp)# import sip domain-name`	Imports a SIP server domain-name list option to the outbound SIP proxy server.
Step 6	**end** **Example:** `Router(config-dhcp)# end`	Returns to privileged EXEC mode.

Configuring the SNTP Server

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ipv6 dhcp pool** *poolname*
4. **sntp address** *ipv6-address*
5. **end**

DETAILED STEPS

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

	Command or Action	Purpose
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**ipv6 dhcp pool** *poolname* **Example:** `Router(config)# ipv6 dhcp pool pool1`	Configures a DHCPv6 configuration information pool and enters DHCPv6 pool configuration mode.
Step 4	**sntp address** *ipv6-address* **Example:** `Router(config-dhcp)# sntp address 2001:DB8:2000:2000::33`	Specifies the SNTP server list to be sent to the client.
Step 5	**end** **Example:** `Router(config-dhcp)# end`	Returns to privileged EXEC mode.

Importing the SNTP Server Option

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ipv6 dhcp pool** *poolname*
4. **import sntp address** *ipv6-address*
5. **end**

DETAILED STEPS

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

	Command or Action	Purpose
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**ipv6 dhcp pool** *poolname* **Example:** `Router(config)# ipv6 dhcp pool pool1`	Configures a DHCPv6 configuration information pool and enters DHCPv6 pool configuration mode.
Step 4	**import sntp address** *ipv6-address* **Example:** `Router(config-dhcp)# import sntp address 2001:DB8:2000:2000::33`	Imports the SNTP server option to a DHCPv6 client.
Step 5	**end** **Example:** `Router(config-dhcp)# end`	Returns to privileged EXEC mode.

Importing Stateless DHCPv6 Server Options

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ipv6 dhcp pool** *poolname*
4. **import dns-server**
5. **import domain-name**
6. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**ipv6 dhcp pool** *poolname* **Example:** `Router(config)# ipv6 dhcp pool pool1`	Configures a DHCPv6 configuration information pool and enters DHCPv6 pool configuration mode.
Step 4	**import dns-server** **Example:** `Router(config-dhcp)# import dns-server`	Imports the DNS recursive name server option to a DHCPv6 client.
Step 5	**import domain-name** **Example:** `Router(config-dhcp)# import domain-name`	Imports the domain search list option to a DHCPv6 client.
Step 6	**end** **Example:** `Router(config-dhcp)# end`	Returns to privileged EXEC mode.

Defining a General Prefix with the DHCPv6 Prefix Delegation Client Function

Perform this task to configure the DHCPv6 client function on an interface and enable prefix delegation on an interface. The delegated prefix is stored in a general prefix.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type number*
4. **ipv6 dhcp client pd** {*prefix-name* | **hint** *ipv6-prefix*} [**rapid-commit**]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**interface** *type number* **Example:** `Router(config)# interface gigabitethernet 0/0/0`	Specifies an interface type and number, and places the router in interface configuration mode.
Step 4	**ipv6 dhcp client pd** {*prefix-name* | **hint** *ipv6-prefix*} [**rapid-commit**] **Example:** `Router(config-if)# ipv6 dhcp client pd dhcp-prefix`	Enables the DHCPv6 client process and enables a request for prefix delegation through a specified interface. • The delegated prefix is stored in the general prefix *prefix-name* argument.

Configuring a VRF-Aware Relay and Server for MPLS VPN Support

Configuring a VRF-Aware Relay

Note You do not have to configure this feature on specified interfaces. If you want the feature to be enabled globally only on the router, perform steps 1, 2, and 3

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ipv6 dhcp-relay option vpn**
4. **interface** *type number*
5. **ipv6 dhcp relay option vpn**
6. **ipv6 dhcp relay destination** *ipv6-address* [*interface-type interface-number* | **vrf** *vrf-name* | **global**]
7. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**ipv6 dhcp-relay option vpn** **Example:** `Router(config)# ipv6 dhcp-relay option vpn`	Enables the DHCP for IPv6 relay VRF-aware feature globally.
Step 4	**interface** *type number* **Example:** `Router(config)# interface gigabitethernet 0/0/0`	Specifies an interface type and number, and enters interface configuration mode.
Step 5	**ipv6 dhcp relay option vpn** **Example:** `Router(config-if)# ipv6 dhcp relay option vpn`	Enables the DHCP for IPv6 relay VRF-aware feature on the specified interface. Enabling this command supersedes the configuration that is enabled by using the **ipv6 dhcp-relay option vpn** command.

	Command or Action	Purpose
Step 6	**ipv6 dhcp relay destination** *ipv6-address* [*interface-type interface-number* | **vrf** *vrf-name* | **global**] **Example:** `Router(config-if)# ipv6 dhcp relay destination FE80::250:A2FF:FEBF:A056 ethernet 0/0`	Specifies a destination address to which client messages are forwarded.
Step 7	**end** **Example:** `Router(config-if)# end`	Returns to privileged EXEC mode.

Configuring a VRF-Aware Server

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type number*
4. **ipv6 dhcp server vrf enable**
5. **end**

DETAILED STEPS

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

	Command or Action	Purpose
Step 3	**interface** *type number* **Example:** `Router(config)# interface gigabitethernet 0/0/0`	Specifies an interface type and number, and enters interface configuration mode.
Step 4	**ipv6 dhcp server vrf enable** **Example:** `Router(config-if)# ipv6 dhcp server vrf enable`	Enables the DHCPv6 server VRF-aware feature on an interface.
Step 5	**end** **Example:** `Router(config-if)# end`	Returns to privileged EXEC mode.

Restarting the DHCPv6 Client on an Interface

Perform this task to restart the DHCPv6 client on a specified interface after first releasing and unconfiguring previously acquired prefixes and other configuration options.

SUMMARY STEPS

1. **enable**
2. **clear ipv6 dhcp client** *interface-type interface-number*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**clear ipv6 dhcp client** *interface-type interface-number* **Example:** `Router# clear ipv6 dhcp client GigabitEthernet 1/0/0`	Restarts DHCPv6 client on an interface.

Deleting Automatic Client Bindings from the DHCPv6 Binding Table

SUMMARY STEPS

1. **enable**
2. **clear ipv6 dhcp binding** [*ipv6-address*] [**vrf** *vrf-name*]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**clear ipv6 dhcp binding** [*ipv6-address*] [**vrf** *vrf-name*] **Example:** `Router# clear ipv6 dhcp binding`	Deletes automatic client bindings from the DHCPv6 binding table.

Troubleshooting DHCPv6

SUMMARY STEPS

1. **enable**
2. **debug ipv6 dhcp** [**detail**]
3. **debug ipv6 dhcp database**
4. **debug ipv6 dhcp relay**

DETAILED STEPS

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

	Command or Action	Purpose
Step 2	**debug ipv6 dhcp** [**detail**] **Example:** `Router# debug ipv6 dhcp`	Enables debugging for DHCPv6.
Step 3	**debug ipv6 dhcp database** **Example:** `Router# debug ipv6 dhcp database`	Enables debugging for the DHCPv6 binding database.
Step 4	**debug ipv6 dhcp relay** **Example:** `Router# debug ipv6 dhcp relay`	Enables DHCPv6 relay agent debugging.

Verifying DHCPv6 Configuration and Operation

SUMMARY STEPS

1. **enable**
2. **show ipv6 dhcp**
3. **show ipv6 dhcp binding** [*ipv6-address*]
4. **show ipv6 dhcp database** [*agent-URL*]
5. **show ipv6 dhcp interface** [*type number*]
6. **show ipv6 dhcp pool** [*poolname*]
7. **show running-config**

DETAILED STEPS

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

	Command or Action	Purpose
Step 2	**show ipv6 dhcp** **Example:** `Router# show ipv6 dhcp`	Displays the DUID on a specified device.
Step 3	**show ipv6 dhcp binding** [*ipv6-address*] **Example:** `Router# show ipv6 dhcp binding`	Displays automatic client bindings from the DHCPv6 database.
Step 4	**show ipv6 dhcp database** [*agent-URL*] **Example:** `Router# show ipv6 dhcp database`	Displays the DHCPv6 binding database agent information.
Step 5	**show ipv6 dhcp interface** [*type number*] **Example:** `Router# show ipv6 dhcp interface`	Displays DHCPv6 interface information.
Step 6	**show ipv6 dhcp pool** [*poolname*] **Example:** `Router# show ipv6 dhcp pool`	Displays DHCPv6 configuration pool information.
Step 7	**show running-config** **Example:** `Router# show running-config`	Displays the current configuration running on the router.

- Examples, page 145

Examples

Sample Output from the show ipv6 dhcp Command

The following sample output from the **show ipv6 dhcp** command shows the DUID of the device:

```
Router# show ipv6 dhcp
This device's DHCPv6 unique identifier(DUID): 000300010002FCA5DC1C
```

Sample Output from the show ipv6 dhcp binding Command

In the following example, the **show ipv6 dhcp binding** command shows information about two clients, including their DUIDs, IAPDs, prefixes, and preferred and valid lifetimes:

```
Router# show ipv6 dhcp binding
Client: FE80::202:FCFF:FEA5:DC39 (GigabitEthernet2/1/0)
  DUID: 000300010002FCA5DC1C
  IA PD: IA ID 0x00040001, T1 0, T2 0
    Prefix: 3FFE:C00:C18:11::/68
            preferred lifetime 180, valid lifetime 12345
            expires at Nov 08 2002 02:24 PM (12320 seconds)
Client: FE80::202:FCFF:FEA5:C039 (GigabitEthernet2/1/0)
  DUID: 000300010002FCA5C01C
  IA PD: IA ID 0x00040001, T1 0, T2 0
    Prefix: 3FFE:C00:C18:1::/72
            preferred lifetime 240, valid lifetime 54321
            expires at Nov 09 2002 02:02 AM (54246 seconds)
    Prefix: 3FFE:C00:C18:2::/72
            preferred lifetime 300, valid lifetime 54333
            expires at Nov 09 2002 02:03 AM (54258 seconds)
    Prefix: 3FFE:C00:C18:3::/72
            preferred lifetime 280, valid lifetime 51111
```

Sample Output from the show ipv6 dhcp database Command

The following sample output from the **show ipv6 dhcp database** command shows information on the binding database agents TFTP, NVRAM, and flash:

```
Router# show ipv6 dhcp database
Database agent tftp://172.19.216.133/db.tftp:
  write delay: 69 seconds, transfer timeout: 300 seconds
  last written at Jan 09 2003 01:54 PM,
     write timer expires in 56 seconds
  last read at Jan 06 2003 05:41 PM
  successful read times 1
  failed read times 0
  successful write times 3172
  failed write times 2
Database agent nvram:/dhcpv6-binding:
  write delay: 60 seconds, transfer timeout: 300 seconds
  last written at Jan 09 2003 01:54 PM,
     write timer expires in 37 seconds
  last read at never
  successful read times 0
  failed read times 0
  successful write times 3325
  failed write times 0
Database agent flash:/dhcpv6-db:
  write delay: 82 seconds, transfer timeout: 3 seconds
  last written at Jan 09 2003 01:54 PM,
    write timer expires in 50 seconds
  last read at never
  successful read times 0
  failed read times 0
  successful write times 2220
  failed write times 614
```

Sample Output from the show ipv6 dhcp interface Command

The following is sample output from the **show ipv6 dhcp interface** command. In the first example, the command is used on a router that has an interface acting as a DHCPv6 server. In the second example, the command is used on a router that has an interface acting as a DHCPv6 client:

```
Router1# show ipv6 dhcp interface
GigabitEthernet2/1/0 is in server mode
  Using pool: svr-p1
  Preference value: 20
```

```
  Rapid-Commit is disabled
Router2# show ipv6 dhcp interface
GigabitEthernet2/1/0 is in client mode
  State is OPEN (1)
  List of known servers:
    Address: FE80::202:FCFF:FEA1:7439, DUID 000300010002FCA17400
    Preference: 20
      IA PD: IA ID 0x00040001, T1 120, T2 192
        Prefix: 3FFE:C00:C18:1::/72
                preferred lifetime 240, valid lifetime 54321
                expires at Nov 08 2002 09:10 AM (54319 seconds)
        Prefix: 3FFE:C00:C18:2::/72
                preferred lifetime 300, valid lifetime 54333
                expires at Nov 08 2002 09:11 AM (54331 seconds)
        Prefix: 3FFE:C00:C18:3::/72
                preferred lifetime 280, valid lifetime 51111
                expires at Nov 08 2002 08:17 AM (51109 seconds)
      DNS server: 2001:DB8:1001::1
      DNS server: 2001:DB8:1001::2
      Domain name: example1.net
      Domain name: example2.net
      Domain name: example3.net
    Prefix name is cli-p1
    Rapid-Commit is enabled
```

Sample Output from the show ipv6 dhcp pool Command

In the following example, the **show ipv6 dhcp pool** command provides information on the configuration pool named svr-p1, including the static bindings, prefix information, the DNS server, and the domain names found in the svr-p1 pool:

```
Router# show ipv6 dhcp pool

DHCPv6 pool: svr-p1
  Static bindings:
    Binding for client 000300010002FCA5C01C
      IA PD: IA ID 00040002,
        Prefix: 3FFE:C00:C18:3::/72
                preferred lifetime 604800, valid lifetime 2592000
      IA PD: IA ID not specified; being used by 00040001
        Prefix: 3FFE:C00:C18:1::/72
                preferred lifetime 240, valid lifetime 54321
        Prefix: 3FFE:C00:C18:2::/72
                preferred lifetime 300, valid lifetime 54333
        Prefix: 3FFE:C00:C18:3::/72
                preferred lifetime 280, valid lifetime 51111
  Prefix from pool: local-p1, Valid lifetime 12345, Preferred lifetime 180
  DNS server: 2001:DB8:1001::1
  DNS server: 2001:DB8:1001::2
  Domain name: example1.net
  Domain name: example2.net
  Domain name: example3.net
Active clients: 2
Current configuration : 22324 bytes
!
! Last configuration change at 14:59:38 PST Tue Jan 16 2009
! NVRAM config last updated at 04:25:39 PST Tue Jan 16 2009 by name01
!
hostname Router
!
ip cef
ipv6 unicast-routing
ipv6 cef
ipv6 cef accounting prefix-length
!
!
interface GigabitEthernet0/0/0
 ip address 10.4.9.11 255.0.0.0
 media-type 10BaseT
 ipv6 address 2001:DB8:C18:1::/64 eui-64
```

Configuration Examples for Implementing DHCPv6

Example: Configuring the DHCPv6 Server Function

DHCPv6 clients are connected to the DHCPv6 server on Gigabit Ethernet interface 0/0/0. The server is configured to use parameters from the DHCP pool called dhcp-pool. This pool provides clients with the IPv6 address of a DNS server and the domain name to be used. It also specifies that prefixes can be delegated from the prefix pool called client-prefix-pool1. The prefixes delegated will have valid and preferred lifetimes of 1800 and 600 seconds respectively. The prefix pool named client-prefix-pool1 has a prefix of length /40 from which it will delegate (sub) prefixes of length /48.

```
ipv6 dhcp pool dhcp-pool
 prefix-delegation pool client-prefix-pool1 lifetime 1800 600
 dns-server 2001:DB8:3000:3000::42
 domain-name example.com
!
interface GigabitEthernet0/0/0
 description downlink to clients
 ipv6 address FEC0:240:104:2001::139/64
 ipv6 dhcp server dhcp-pool
!
ipv6 local pool client-prefix-pool1 2001:DB8:1200::/40 48
```

The following example from the **show ipv6 dhcp** command shows the DUID of the device:

```
Router# show ipv6 dhcp

This device's DHCPv6 unique identifier(DUID): 000300010002FCA5DC1C
```

In the following example, the show ipv6 dhcp binding command shows information about two clients, including their DUIDs, IAPDs, prefixes, and preferred and valid lifetimes:

```
Router# show ipv6 dhcp binding

Client: FE80::202:FCFF:FEA5:DC39 (GigabitEthernet2/1/0)
  DUID: 000300010002FCA5DC1C
  IA PD: IA ID 0x00040001, T1 0, T2 0
    Prefix: 3FFE:C00:C18:11::/68
            preferred lifetime 180, valid lifetime 12345
            expires at Nov 08 2002 02:24 PM (12320 seconds)
Client: FE80::202:FCFF:FEA5:C039 (GigabitEthernet2/1/0)
  DUID: 000300010002FCA5C01C
  IA PD: IA ID 0x00040001, T1 0, T2 0
    Prefix: 3FFE:C00:C18:1::/72
            preferred lifetime 240, valid lifetime 54321
            expires at Nov 09 2002 02:02 AM (54246 seconds)
    Prefix: 3FFE:C00:C18:2::/72
            preferred lifetime 300, valid lifetime 54333
            expires at Nov 09 2002 02:03 AM (54258 seconds)
    Prefix: 3FFE:C00:C18:3::/72
            preferred lifetime 280, valid lifetime 51111
```

In the following example, the **show ipv6 dhcp database** command provides information on the binding database agents TFTP, NVRAM, and flash:

```
Router# show ipv6 dhcp database

Database agent tftp://172.19.216.133/db.tftp:
  write delay: 69 seconds, transfer timeout: 300 seconds
```

```
  last written at Jan 09 2003 01:54 PM,
     write timer expires in 56 seconds
  last read at Jan 06 2003 05:41 PM
  successful read times 1
  failed read times 0
  successful write times 3172
  failed write times 2
Database agent nvram:/dhcpv6-binding:
  write delay: 60 seconds, transfer timeout: 300 seconds
  last written at Jan 09 2003 01:54 PM,
     write timer expires in 37 seconds
  last read at never
  successful read times 0
  failed read times 0
  successful write times 3325
  failed write times 0
Database agent flash:/dhcpv6-db:
  write delay: 82 seconds, transfer timeout: 3 seconds
  last written at Jan 09 2003 01:54 PM,
    write timer expires in 50 seconds
  last read at never
  successful read times 0
  failed read times 0
  successful write times 2220
  failed write times 614
```

Example: Configuring the DHCPv6 Client Function

This DHCPv6 client has three interfaces: Gigabit Ethernet interface 0/0/0 is the upstream link to a service provider, which has a DHCPv6 server function enabled. The Fast Ethernet interfaces 0/0/0 and 0/1/0 are links to local networks.

The upstream interface, Gigabit Ethernet interface 0/0/0, has the DHCPv6 client function enabled. Prefixes delegated by the provider are stored in the general prefix called prefix-from-provider.

The local networks, Fast Ethernet interfaces 0/0/0 and 0/1/0, both assign interface addresses based on the general prefix called prefix-from-provider. The bits on the left of the addresses come from the general prefix, and the bits on the right of the addresses are specified statically.

```
interface GigabitEthernet 0/0/0
 description uplink to provider DHCP IPv6 server
 ipv6 dhcp client pd prefix-from-provider
!
interface FastEthernet 0/0/0
 description local network 0
 ipv6 address prefix-from-provider ::5:0:0:0:100/64
!
interface FastEthernet 0/1/0
 description local network 1
 ipv6 address prefix-from-provider ::6:0:0:0:100/64
```

Example: Configuring a Database Agent for the Server Function

The DHCPv6 server is configured to store table bindings to the file named dhcp-binding on the server at address 10.0.0.1 using the TFTP protocol. The bindings are saved every 120 seconds.

```
ipv6 dhcp database tftp://10.0.0.1/dhcp-binding write-delay 120
```

The following example shows how to specify DHCP for IPv6 binding database agent parameters and store binding entries in bootflash:

```
ipv6 dhcp database bootflash
```

Example: Configuring the Stateless DHCPv6 Function

The following example shows how to use the DHCPv6 function to configure clients with information about the name lookup system. The server is configured with a DHCP pool, which contains the name lookup information that is to be passed to clients. It does not need to contain a prefix pool. This DHCP pool is attached to the access link to customers (GigabitEthernet0/0/0) using the **ipv6 dhcp server** command. The access link also has the **ipv6 nd other-config-flag** command enabled. RA messages sent from this interface will inform clients that they should use DHCPv6 for "other" (for example, nonaddress) configuration information.

```
ipv6 dhcp pool dhcp-pool
 dns-server 2001:DB8:A:B::1
 dns-server 2001:DB8:3000:3000::42
 domain-name example.com
!
interface GigabitEthernet0/0/0
description Access link down to customers
 ipv6 address 2001:DB8:1234:42::1/64
 ipv6 nd other-config-flag
 ipv6 dhcp server dhcp-pool
```

The client has no obvious DHCPv6 configuration. However, the **ipv6 address autoconfig** command on the uplink to the service provider (GigabitEthernet 0/0/0) causes the following two events:

- Addresses are autoconfigured on the interface, based on prefixes in RA messages received from the server.
- If received RA messages have the "other configuration" flag set, the interface will attempt to acquire the other (for example, nonaddress) configuration from any DHCPv6 servers.

Additional References

Related Documents

Related Topic	Document Title
IPv6 supported feature list	"Start Here: Cisco IOS XE Software Release Specifics for IPv6 Features ," *Cisco IOS XE IPv6 Configuration Guide*
IPv6 basic connectivity	" Implementing IPv6 Addressing and Basic Connectivity ," *Cisco IOS XE IPv6 Configuration Guide*
IPv6 prefix delegation	• " Implementing IPv6 Addressing and Basic Connectivity ," *Cisco IOS XE IPv6 Configuration Guide* • " Implementing ADSL and Deploying Dial Access for IPv6 ," *Cisco IOS XE IPv6 Configuration Guide*

Related Topic	Document Title
IPv6 commands: complete command syntax, command mode, defaults, usage guidelines, and examples	*Cisco IOS IPv6 Command Reference*
Cisco IOS master command list, all releases	Cisco IOS Master Command List, All Releases

Standards and RFCs

Standards/RFCs	Title
RFC 3315	*Dynamic Host Configuration Protocol for IPv6*
RFC 3319	*Dynamic Host Configuration Protocol (DHCPv6) Options for Session Initiation Protocol (SIP) Servers*
RFC 3633	*IPv6 Prefix Options for Dynamic Host Configuration Protocol (DHCP) Version 6*
RFC 3646	*DNS Configuration Options for Dynamic Host Configuration Protocol for IPv6 (DHCPv6)*
RFC 3898	*Network Information Service (NIS) Configuration Options for Dynamic Host Configuration Protocol for IPv6 (DHCPv6)*
RFC 4075	*Simple Network Time Protocol (SNTP) Configuration Option for DHCPv6*
RFC 4242	*Information Refresh Time Option for Dynamic Host Configuration Protocol for IPv6 (DHCPv6)*
RFC 4649	*Dynamic Host Configuration Protocol for IPv6 (DHCPv6) Relay Agent Remote-ID Option*

MIBs

MIBs	MIBs Link
None	To locate and download MIBs for selected platforms, Cisco software 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 and Documentation website provides online resources to download documentation, software, and tools. Use these resources to install and configure the software and to troubleshoot and resolve technical issues with Cisco products and technologies. Access to most tools on the Cisco Support and Documentation website requires a Cisco.com user ID and password.	http://www.cisco.com/cisco/web/support/index.html

Feature Information for Implementing DHCP for IPv6

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Table 9 Feature Information for Implementing DHCP for IPv6

Feature Name	Releases	Feature Information
IPv6 Access Services—DHCPv6 Prefix Delegation	Cisco IOS XE Release 2.1	The DHCPv6 prefix delegation feature can be used to manage link, subnet, and site addressing changes. DHCPv6 can be used in environments to deliver stateful and stateless information. The following commands were modified by this feature: **clear ipv6 dhcp binding**, **clear ipv6 dhcp client**, **debug ipv6 dhcp**, **debug ipv6 dhcp database**, **dns-server (IPv6)**, **domain-name (IPv6)**, **ipv6 dhcp client pd**, **ipv6 dhcp database**, **ipv6 dhcp pool**, **ipv6 dhcp server**, **prefix-delegation**, **prefix-delegation pool**, **show ipv6 dhcp**, **show ipv6 dhcp binding**, **show ipv6 dhcp database**, **show ipv6 dhcp interface**, **show ipv6 dhcp pool**

Feature Name	Releases	Feature Information
IPv6 Access Services—Stateless DHCPv6	Cisco IOS XE Release 2.5	The stateless DHCPv6 feature allows DHCPv6 to be used for configuring a node with parameters that do not require a server to maintain any dynamic state for the node. • The following commands were modified by this feature: **dns-server**, **domain-name**, **ipv6 dhcp pool**, **import dns-server**, **import domain-name**, **import information refresh**, **import nis address**, **import nisdomain-name**, **import nisp address**, **import nisp domain-name**, **import sip address**, **import sip domain-name**, **import sntp address**, **information refresh, ipv6 dhcp server**, **ipv6 nd other-config-flag**, **nis address**, **nis domain-name**, **nisp address**, **nisp domain-name**, **show ipv6 dhcp interface**, **show ipv6 dhcp pool,**, **sntp address**
IPv6 Access Services—DHCP for IPv6 Relay Agent	Cisco IOS XE Release 2.1	A DHCP relay agent, which may reside on the client's link, is used to relay messages between the client and server. The following commands were modified by this feature: **debug ipv6 dhcp relay**, **ipv6 dhcp relay destination**, **show ipv6 dhcp interface**
DHCP—DHCPv6 Relay Agent Notification for Prefix Delegation	Cisco IOS XE Release 2.1	DHCPv6 relay agent notification for prefix delegation allows the router working as a DHCPv6 relay agent to find prefix delegation options by reviewing the contents of a DHCPv6 packet that is being relayed by the relay agent to the client.

Feature Name	Releases	Feature Information
DHCPv6 Ethernet Remote ID Option	Cisco IOS XE Release 2.1	This feature adds the remote-ID option to relayed (RELAY-FORWARD) DHCPv6 packets.
DHCPv6—Relay chaining (for Prefix Delegation) and route insertion in FIB	Cisco IOS XE Release 3.5S	This feature allows DHCPv6 messages to be relayed through multiple relay agents. The following commands were introduced or modified by this feature: **clear ipv6 dhcp relay binding**, **clear ipv6 dhcp route**, **ipv6 dhcp iana-route-add** , **ipv6 dhcp iapd-route-add**, **show ipv6 dhcp relay binding**, **show ipv6 dhcp route.**
DHCPv6 - Relay - Reload Persistent Interface ID Option	Cisco IOS XE Release 2.1	This feature makes the interface-ID option, which is used by relay agents to decide which interface should be used when forwarding a RELAY-REPLY packet, persistent.
DHCPv6 Server—MPLS VPN Support	Cisco IOS XE Release 3.3S	The DHCPv6 server implementation of MPLS VPN support allows a per-pool configuration so DHCPv6 pools can be associated with a VRF instance. The DHCPv6 relay implementation allows the configuration of the destination VRF instance to which the relay messages will be forwarded.

Implementing EIGRP for IPv6

Customers can configure Enhanced Interior Gateway Routing Protocol (EIGRP) to route IPv6 prefixes. EIGRP IPv4 runs over an IPv4 transport, communicates only with IPv4 peers, and advertises only IPv4 routes, and EIGRP for IPv6 follows the same model. EIGRP for IPv4 and EIGRP for IPv6 are configured and managed separately. However, the configuration of EIGRP for IPv4 and IPv6 is similar and provides operational familiarity and continuity.

This document provides information about configuring and implementing EIGRP for IPv6.

Finding Feature Information

Your software release may not support all the features documented in this module. For the latest caveats and feature information, see Bug Search Tool and the release notes for your platform and software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the feature information table at the end of this module.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Restrictions for Implementing EIGRP for IPv6

This section lists ways in which EIGRP for IPv6 differs from EIGRP IPv4 and lists EIGRP for IPv6 restrictions:

- EIGRP for IPv6 is directly configured on the interfaces over which it runs. This feature allows EIGRP for IPv6 to be configured without the use of a global IPv6 address. There is no network statement in EIGRP for IPv6.

In per-interface configuration at system startup, if EIGRP has been configured on an interface, then the EIGRP protocol may start running before any EIGRP router mode commands have been executed.

- An EIGRP for IPv6 protocol instance requires a router ID before it can start running.

- EIGRP for IPv6 has a shutdown feature. The routing process should be in "no shut" mode in order to start running.
- When a user uses a passive-interface configuration, EIGRP for IPv6 need not be configured on the interface that is made passive.
- EIGRP for IPv6 provides route filtering using the distribute-list prefix-list command. Use of the route-map command is not supported for route filtering with a distribute list.

Information About Implementing EIGRP for IPv6

Cisco EIGRP for IPv6 Implementation

EIGRP is an enhanced version of the IGRP developed by Cisco. EIGRP uses the same distance vector algorithm and distance information as IGRP. However, the convergence properties and the operating efficiency of EIGRP have improved substantially over IGRP.

The convergence technology is based on research conducted at SRI International and employs an algorithm called the diffusing update algorithm (DUAL). This algorithm guarantees loop-free operation at every instant throughout a route computation and allows all devices involved in a topology change to synchronize at the same time. Devices that are not affected by topology changes are not involved in recomputations. The convergence time with DUAL rivals that of any other existing routing protocol.

EIGRP provides the following features:

- Increased network width--With Routing Information Protocol (RIP), the largest possible width of your network is 15 hops. When EIGRP is enabled, the largest possible width is 224 hops. Because the EIGRP metric is large enough to support thousands of hops, the only barrier to expanding the network is the transport layer hop counter. Cisco works around this limitation by incrementing the transport control field only when an IPv4 or an IPv6 packet has traversed 15 devices and the next hop to the destination was learned by way of EIGRP. When a RIP route is being used as the next hop to the destination, the transport control field is incremented as usual.
- Fast convergence--The DUAL algorithm allows routing information to converge as quickly as any other routing protocol.
- Partial updates--EIGRP sends incremental updates when the state of a destination changes, instead of sending the entire contents of the routing table. This feature minimizes the bandwidth required for EIGRP packets.
- Neighbor discovery mechanism--This is a simple hello mechanism used to learn about neighboring devices. It is protocol-independent.
- Arbitrary route summarization.
- Scaling--EIGRP scales to large networks.
- Route filtering--EIGRP for IPv6 provides route filtering using the **distribute-list prefix-list**command. Use of the **route-map** command is not supported for route filtering with a distribute list.

EIGRP has the following four basic components:

- Neighbor discovery--Neighbor discovery is the process that devices use to dynamically learn of other devices on their directly attached networks. Devices must also discover when their neighbors become unreachable or inoperative. EIGRP neighbor discovery is achieved with low overhead by periodically sending small hello packets. EIGRP neighbors can also discover a neighbor that has recovered after an outage because the recovered neighbor will send out a hello packet. As long as hello packets are

received, the Cisco software can determine that a neighbor is alive and functioning. Once this status is determined, the neighboring devices can exchange routing information.

- Reliable transport protocol--The reliable transport protocol is responsible for guaranteed, ordered delivery of EIGRP packets to all neighbors. It supports intermixed transmission of multicast and unicast packets. Some EIGRP packets must be sent reliably and others need not be. For efficiency, reliability is provided only when necessary. For example, on a multiaccess network that has multicast capabilities, it is not necessary to send hello packets reliably to all neighbors individually. Therefore, EIGRP sends a single multicast hello with an indication in the packet informing the receivers that the packet need not be acknowledged. Other types of packets (such as updates) require acknowledgment, which is indicated in the packet. The reliable transport has a provision to send multicast packets quickly when unacknowledged packets are pending. This provision helps to ensure that convergence time remains low in the presence of varying speed links.
- DUAL finite state machine--The DUAL finite state machine embodies the decision process for all route computations. It tracks all routes advertised by all neighbors. DUAL uses several metrics including distance and cost information to select efficient, loop-free paths. When multiple routes to a neighbor exist, DUAL determines which route has the lowest metric (named the feasible distance), and enters this route into the routing table. Other possible routes to this neighbor with larger metrics are received, and DUAL determines the reported distance to this network. The reported distance is defined as the total metric advertised by an upstream neighbor for a path to a destination. DUAL compares the reported distance with the feasible distance, and if the reported distance is less than the feasible distance, DUAL considers the route to be a feasible successor and enters the route into the topology table. The feasible successor route that is reported with the lowest metric becomes the successor route to the current route if the current route fails. To avoid routing loops, DUAL ensures that the reported distance is always less than the feasible distance for a neighbor device to reach the destination network; otherwise, the route to the neighbor may loop back through the local device.
- Protocol-dependent modules--When there are no feasible successors to a route that has failed, but there are neighbors advertising the route, a recomputation must occur. This is the process in which DUAL determines a new successor. The amount of time required to recompute the route affects the convergence time. Recomputation is processor-intensive; it is advantageous to avoid unneeded recomputation. When a topology change occurs, DUAL will test for feasible successors. If there are feasible successors, DUAL will use them in order to avoid unnecessary recomputation.

The protocol-dependent modules are responsible for network layer protocol-specific tasks. For example, the EIGRP module is responsible for sending and receiving EIGRP packets that are encapsulated in IPv4 or IPv6. It is also responsible for parsing EIGRP packets and informing DUAL of the new information received. EIGRP asks DUAL to make routing decisions, but the results are stored in the IPv4 or IPv6 routing table. Also, EIGRP is responsible for redistributing routes learned by other IPv4 or IPv6 routing protocols.

How to Implement EIGRP for IPv6

Enabling EIGRP for IPv6 on an Interface

EIGRP for IPv6 is directly configured on the interfaces over which it runs, which allows EIGRP for IPv6 to be configured without the use of a global IPv6 address.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ipv6 unicast-routing**
4. **interface** *type number*
5. **no shut**
6. **ipv6 enable**
7. **ipv6 eigrp** *as-number*
8. **ipv6 router eigrp** *as-number*
9. **router-id** {*ip-address* | *ipv6-address*
10. **exit**
11. **show ipv6 eigrp** [*as-number*] **interfaces** [*type number*] [*as-number*]

DETAILED STEPS

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

	Command or Action	Purpose
Step 4	**interface** *type number* **Example:** `Device(config)# interface GigabitEthernet 0/0/0`	Specifies the interface on which EIGRP is to be configured.
Step 5	**no shut** **Example:** `Device(config-if)# no shut`	Enables no shut mode so the routing process can start running.
Step 6	**ipv6 enable** **Example:** `Device(config-if)# ipv6 enable`	Enables IPv6 processing on an interface that has not been configured with an explicit IPv6 address.
Step 7	**ipv6 eigrp** *as-number* **Example:** `Device(config-if)# ipv6 eigrp 1`	Enables EIGRP for IPv6 on a specified interface.
Step 8	**ipv6 router eigrp** *as-number* **Example:** `Device(config-if)# ipv6 router eigrp 1`	Enters router configuration mode and creates an EIGRP IPv6 routing process.
Step 9	**router-id** {*ip-address* | *ipv6-address* **Example:** `Device(config-router)# router-id 10.1.1.1`	Enables the use of a fixed router ID. Use this command only if an IPv4 address is not defined on the router eligible for router ID.
Step 10	**exit** **Example:** `Device(config-router) exit`	Enter three times to return to privileged EXEC mode.

	Command or Action	Purpose
Step 11	**show ipv6 eigrp** [*as-number*] **interfaces** [*type number*] [*as-number*] **Example:** `Device# show ipv6 eigrp interfaces`	Displays information about interfaces configured for EIGRP for IPv6.

Configuring the Percentage of Link Bandwidth Used by EIGRP

By default, EIGRP packets consume a maximum of 50 percent of the link bandwidth, as configured with the **bandwidth interface** command. You might want to change that value if a different level of link utilization is required or if the configured bandwidth does not match the actual link bandwidth (it may have been configured to influence route metric calculations).

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type number*
4. **no shut**
5. **ipv6 bandwidth-percent eigrp** *as-number percent*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *type number* **Example:** `Device(config)# interface GigabitEthernet 0/0/0`	Specifies the interface on which EIGRP is configured.

	Command or Action	Purpose
Step 4	**no shut** **Example:** `Device(config)# no shut`	Enables no shut mode so the routing process can start running.
Step 5	**ipv6 bandwidth-percent eigrp** *as-number percent* **Example:** `Device(config-if)# ipv6 bandwidth-percent eigrp 1 75`	Configures the percentage of bandwidth that may be used by EIGRP for IPv6 on an interface

Configuring Summary Addresses

If any more specific routes are in the routing table, EIGRP for IPv6 will advertise the summary address out the interface with a metric equal to the minimum of all more specific routes.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type number*
4. **no shut**
5. **ipv6 summary-address eigrp** *as-number ipv6-address* [*admin-distance*]

DETAILED STEPS

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

	Command or Action	Purpose
Step 3	**interface** *type number* **Example:** `Device(config)# interface GigabitEthernet 0/0/0`	Specifies the interface on which EIGRP is configured.
Step 4	**no shut** **Example:** `Device(config)# no shut`	Enables no shut mode so the routing process can start running.
Step 5	**ipv6 summary-address eigrp** *as-number ipv6-address* [*admin-distance*] **Example:** `Device(config-if)# ipv6 summary-address eigrp 1 2001:DB8:0:1::/64`	Configures a summary aggregate address for a specified interface.

Configuring EIGRP Route Authentication

EIGRP route authentication provides message digest algorithm 5 (MD5) authentication of routing updates from the EIGRP routing protocol. The MD5 keyed digest in each EIGRP packet prevents the introduction of unauthorized or false routing messages from unapproved sources.

Each key has its own key identifier, which is stored locally. The combination of the key identifier and the interface associated with the message uniquely identifies the authentication algorithm and MD5 authentication key in use.

You can configure multiple keys with lifetimes. Only one authentication packet is sent, regardless of how many valid keys exist. The software examines the key numbers in order from lowest to highest, and uses the first valid key it encounters. Note that the router needs to know the time.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type number*
4. **no shut**
5. **ipv6 authentication mode eigrp** *as-number* **md5**
6. **ipv6 authentication key-chain eigrp** *as-number key-chain*
7. **exit**
8. **key chain** *name-of-chain*
9. **key** *key-id*
10. **key-string** *text*
11. **accept-lifetime** *start-time* **infinite** | *end-time*| **duration** *seconds*
12. **send-lifetime** *start-time* **infinite** | *end-time* | **duration** *seconds*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *type number* **Example:** `Device(config)# interface GigabitEthernet 0/0/0`	Specifies the interface on which EIGRP is configured.
Step 4	**no shut** **Example:** `Device(config)# no shut`	Enables no shut mode so the routing process can start running.

	Command or Action	Purpose
Step 5	**ipv6 authentication mode eigrp** *as-number* **md5** **Example:** `Device(config-if)# ipv6 authentication mode eigrp 1 md5`	Specifies the type of authentication used in EIGRP for IPv6 packets.
Step 6	**ipv6 authentication key-chain eigrp** *as-number key-chain* **Example:** `Device(config-if)# ipv6 authentication key-chain eigrp 1 chain1`	Enables authentication of EIGRP for IPv6 packets.
Step 7	**exit** **Example:** `Device(config-if)# exit`	Exits to global configuration mode.
Step 8	**key chain** *name-of-chain* **Example:** `Device(config)# key chain chain1`	Identifies a group of authentication keys. • Use the name specified in Step 5.
Step 9	**key** *key-id* **Example:** `Device(config-keychain)# key 1`	Identifies an authentication key on a key chain.
Step 10	**key-string** *text* **Example:** `Device(config-keychain-key)# key-string chain 1`	Specifies the authentication string for a key.
Step 11	**accept-lifetime** *start-time* **infinite** | *end-time*| **duration** *seconds* **Example:** `Device(config-keychain-key)# accept-lifetime 14:30:00 Jan 10 2006 duration 7200`	Sets the time period during which the authentication key on a key chain is received as valid.

	Command or Action	Purpose
Step 12	**send-lifetime** *start-time* **infinite** | *end-time* | **duration** *seconds* **Example:** `Device(config-keychain-key)# send-lifetime 15:00:00 Jan 10 2006 duration 3600`	Sets the time period during which an authentication key on a key chain is valid to be sent.

Overriding the Next Hop in EIGRP

EIGRP will, by default, set the IPv6 next-hop value to be itself for routes that it is advertising, even when advertising those routes back out the same interface where it learned them. Perform this task to change this default and instruct EIGRP to use the received next-hop value when advertising these routes.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type number*
4. **no shut**
5. **no ipv6 next-hop-self eigrp** *as-number*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *type number* **Example:** `Device(config)# interface GigabitEthernet 0/0/0`	Specifies the interface on which EIGRP is configured.

	Command or Action	Purpose
Step 4	**no shut** **Example:** `Device(config)# no shut`	Enables no shut mode so the routing process can start running.
Step 5	**no ipv6 next-hop-self eigrp** *as-number* **Example:** `Device(config-if)# no ipv6 next-hop-self eigrp 1`	Changes the default IPv6 next-hop value and instructs EIGRP to use the received next-hop value.

Adjusting the Interval Between Hello Packets in EIGRP for IPv6

Routing devices periodically send hello packets to each other to dynamically learn of other routers on their directly attached networks. This information is used to discover neighbors and to learn when neighbors become unreachable or inoperative.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type number*
4. **ipv6 hello-interval eigrp** *as-number seconds*

DETAILED STEPS

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

	Command or Action	Purpose
Step 3	**interface** *type number* **Example:** `Device(config)# interface GigabitEthernet 0/0/0`	Specifies the interface on which EIGRP is configured.
Step 4	**ipv6 hello-interval eigrp** *as-number seconds* **Example:** `Device(config)# ipv6 hello-interval eigrp 1 10`	Configures the hello interval for the EIGRP for IPv6 routing process designated by an autonomous system number.

Adjusting the Hold Time in EIGRP for IPv6

On very congested and large networks, the default hold time might not be sufficient time for all routers to receive hello packets from their neighbors. In this case, you may want to increase the hold time.

Perform this task to configure the hold time on a specified interface for a particular EIGRP routing process designated by the autonomous system number. The hold time is advertised in hello packets and indicates to neighbors the length of time they should consider the sender valid. The default hold time is three times the hello interval, or 15 seconds. For slow-speed nonbroadcast multi-access (NBMA) networks, the default hold time is 180 seconds. The hold time should be changed if the hello-interval value is changed.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type number*
4. **no shut**
5. **ipv6 hold-time eigrp** *as-number seconds*

DETAILED STEPS

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

	Command or Action	Purpose
Step 3	**interface** *type number* **Example:** `Device(config)# interface GigabitEthernet 0/0/0`	Specifies the interface on which EIGRP is configured.
Step 4	**no shut** **Example:** `Device(config)# no shut`	Enables no shut mode so the routing process can start running.
Step 5	**ipv6 hold-time eigrp** *as-number seconds* **Example:** `Device(config)# ipv6 hold-time eigrp 1 40`	Configures the hold time for a particular EIGRP for IPv6 routing process designated by the autonomous system number.

Disabling Split Horizon in EIGRP for IPv6

By default, split horizon is enabled on all interfaces. Split horizon controls the sending of EIGRP update and query packets. When split horizon is enabled on an interface, update and query packets are not sent for destinations for which this interface is the next hop. Controlling update and query packets in this manner reduces the possibility of routing loops.

Split horizon blocks route information from being advertised by a router out of any interface from which that information originated. This behavior usually optimizes communications among multiple routing devices, particularly when links are broken. However, with nonbroadcast networks (such as multipoint GRE), situations can arise for which this behavior is not ideal. For these situations, including networks in which you have EIGRP configured, you may want to disable split horizon.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type number*
4. **no shut**
5. **no ipv6 split-horizon eigrp** *as-number*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *type number* **Example:** `Device(config)# interface GigabitEthernet 0/0/0`	Specifies the interface on which EIGRP is configured.
Step 4	**no shut** **Example:** `Device(config)# no shut`	Enables no shut mode so the routing process can start running.
Step 5	**no ipv6 split-horizon eigrp** *as-number* **Example:** `Device(config-if)# no ipv6 split-horizon eigrp 101`	Disables EIGRP for IPv6 split horizon on the specified interface.

Configuring EIGRP Stub Routing for Greater Network Stability

The EIGRP stub routing feature can help to provide greater network stability. In the event of network instability, this feature prevents EIGRP queries from being sent over limited bandwidth links to nontransit devices. Instead, distribution devices to which the stub device is connected answer the query on behalf of the stub device. This feature greatly reduces the chance of further network instability due to congested or problematic WAN links. The EIGRP stub routing feature also simplifies the configuration and maintenance of hub-and-spoke networks. When stub routing is enabled in dual-homed remote configurations, it is no longer necessary to configure filtering on remote devices to prevent those remote devices from appearing as transit paths to the hub devices.

Caution EIGRP stub routing should be used only on stub devices. A stub device is defined as a device connected to the network core or distribution layer through which core transit traffic should not flow. A stub device should not have any EIGRP neighbors other than distribution devices.

Configuring a Router for EIGRP Stub Routing

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type number*
4. **no shut**
5. **ipv6 router eigrp** *as-number*
6. **eigrp stub receive-only** | **leak-map** | **connected** | **static** | **summary** | **redistributed**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**interface** *type number* **Example:** `Router(config)# interface GigabitEthernet 0/0/0`	Specifies the interface on which EIGRP is configured.
Step 4	**no shut** **Example:** `Router(config)# no shut`	Enables no shut mode so the routing process can start running.

	Command or Action	Purpose
Step 5	**ipv6 router eigrp** *as-number* **Example:** `Router(config-if)# ipv6 router eigrp 1`	Specifies the EIGRP for IPv6 routing process to be configured.
Step 6	**eigrp stub receive-only** | **leak-map** | **connected** | **static** | **summary** | **redistributed** **Example:** `Router(config-router)# eigrp stub`	Configures a router as a stub using EIGRP.

Verifying EIGRP Stub Routing

SUMMARY STEPS

1. **enable**
2. **show ipv6 eigrp neighbors detail** *interface-type* | *as-number* | **static**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**show ipv6 eigrp neighbors detail** *interface-type* | *as-number* | **static** **Example:** `Device# show ipv6 eigrp neighbors detail`	Displays the neighbors discovered by EIGRP for IPv6. This command is performed on the distribution layer device to view the status of the remote device.

Customizing an EIGRP for IPv6 Routing Process

Logging EIGRP Neighbor Adjacency Changes

You can enable the logging of neighbor adjacency changes to monitor the stability of the routing system and to help you detect problems. By default, adjacency changes are logged.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ipv6 router eigrp** *as-number*
4. **eigrp log-neighbor-changes**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**ipv6 router eigrp** *as-number* **Example:** `Device(config)# ipv6 router eigrp 1`	Specifies the EIGRP for IPv6 routing process to be configured.
Step 4	**eigrp log-neighbor-changes** **Example:** `Device(config-router)# eigrp log-neighbor-changes`	Enables the logging of changes in EIGRP for IPv6 neighbor adjacencies.

Configuring Intervals Between Neighbor Warnings

When neighbor warning messages occur, they are logged by default.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ipv6 router eigrp** *as-number*
4. **eigrp log-neighbor-warnings** [*seconds*]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**ipv6 router eigrp** *as-number* **Example:** `Device(config)# ipv6 router eigrp 1`	Specifies the EIGRP for IPv6 routing process to be configured.
Step 4	**eigrp log-neighbor-warnings** [*seconds*] **Example:** `Device(config-router)# eigrp log-neighbor-warnings 300`	Configures the logging intervals of EIGRP neighbor warning messages.

Adjusting EIGRP for IPv6 Metric Weights

EIGRP for IPv6 uses the minimum bandwidth on the path to a destination network and the total delay to compute routing metrics. You can use the **metric weights** command to adjust the default behavior of EIGRP for IPv6 routing and metric computations. EIGRP for IPv6 metric defaults have been carefully selected to provide optimal performance in most networks.

Note Adjusting EIGRP metric weights can dramatically affect network performance. Because of the complexity of this task, we recommend that you do not change the default values without guidance from an experienced network designer.

By default, the EIGRP composite metric is a 32-bit quantity that is a sum of the segment delays and the lowest segment bandwidth (scaled and inverted) for a given route. For a network of homogeneous media, this metric reduces to a hop count. For a network of mixed media (e.g., GigabitEthernet, FastEthernet, Ethernet), the route with the lowest metric reflects the most desirable path to a destination.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ipv6 router eigrp** *as-number*
4. **metric weights** *tos k1 k2 k3 k4 k5*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**ipv6 router eigrp** *as-number* **Example:** `Device(config)# ipv6 router eigrp 1`	Specifies the EIGRP for IPv6 routing process to be configured.
Step 4	**metric weights** *tos k1 k2 k3 k4 k5* **Example:** `Device(config-router)# metric weights 0 2 0 2 0 0`	Tunes EIGRP metric calculations.

Monitoring and Maintaining EIGRP

SUMMARY STEPS

1. **enable**
2. **clear ipv6 eigrp** [*as-number*] [**neighbor** [*ipv6-address* | *interface-type interface-number*]]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**clear ipv6 eigrp** [*as-number*] [**neighbor** [*ipv6-address* | *interface-type interface-number*]] **Example:** `Router# clear ipv6 eigrp neighbor 3FEE:12E1:2AC1:EA32`	Deletes entries from EIGRP for IPv6 routing tables. The routes that are cleared are the routes that were learned by the specified router.

Configuration Examples for Implementing EIGRP for IPv6

Example Configuring EIGRP to Establish Adjacencies on an Interface

EIGRP for IPv6 is configured directly on the interfaces over which it runs. This example shows the minimal configuration required for EIGRP for IPv6 to send hello packets in order to establish adjacencies on GigabitEthernet 0/0/0:

```
ipv6 unicast-routing
interface gigabitethernet0/0/0
  ipv6 enable
  ipv6 eigrp 1
  no shut
!
ipv6 router eigrp 1
  router-id 10.1.1.1
```

Additional References

Related Documents

Related Topic	Document Title
EIGRP for IPv6 commands	*Cisco IOS IPv6 Command Reference*
IPv6 supported feature list	Start Here: Cisco IOS Software Release Specifics for IPv6 Features

Related Topic	Document Title
Implementing IS-IS for IPv6	Implementing IS-IS for IPv6
Implementing Multiprotocol BGP for IPv6	Implementing Multiprotocol BGP for IPv6
Implementing RIP for IPv6	Implementing RIP for IPv6
EIGRP for IPv4	" Configuring EIGRP ," *Cisco IOS IP Routing Protocols Configuration Guide*
EIGRP for IPv4 commands	" EIGRP Commands ," *Cisco IOS IP Routing Protocols Command Reference*
IPv6 and IPv4 commands for all releases	Cisco IOS Master Command List

Standards

Standard	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
None	To locate and download MIBs for selected platforms, Cisco software releases, and feature sets, use Cisco MIB Locator found at the following URL: http://www.cisco.com/go/mibs

RFCs

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

Feature Information for Implementing EIGRP for IPv6

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Table 10 Feature Information for Implementing EIGRP for IPv6

Feature Name	Releases	Feature Information
IPv6 Routing--EIGRP Support	Cisco IOS XE Release 2.1	Customers can configure EIGRP to route IPv6 prefixes. There is no linkage between EIGRP for IPv4 and EIGRP for IPv6; they are configured and managed separately. However, the configuration of EIGRP for IPv4 and IPv6 is similar and provides operational familiarity and continuity.

Configuring First Hop Redundancy Protocols in IPv6

IPv6 routing protocols ensure router-to-router resilience and failover. However, in situations in which the path between a host and the first-hop router fails, or the first-hop router itself fails, first hop redundancy protocols (FHRPs) ensure host-to-router resilience and failover.

The Hot Standby Router Protocol (HSRP) protects data traffic in case of a gateway failure.

Finding Feature Information

Your software release may not support all the features documented in this module. For the latest caveats and feature information, see Bug Search Tool and the release notes for your platform and software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the feature information table at the end of this module.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Prerequisites for First Hop Redundancy Protocols in IPv6

HSRP version 2 must be enabled on an interface before HSRP IPv6 can be configured.

Information About First Hop Redundancy Protocols in IPv6

HSRP for IPv6

IPv6 routing protocols ensure router-to-router resilience and failover. However, in situations in which the path between a host and the first-hop router fails, or the first-hop router itself fails, first hop redundancy protocols (FHRPs) ensure host-to-router resilience and failover.

The Hot Standby Router Protocol (HSRP) protects data traffic in case of a gateway failure.

HSRP for IPv6 Overview

The HSRP is an FHRP designed to allow for transparent failover of the first-hop IP router. HSRP provides high network availability by providing first-hop routing redundancy for IP hosts on GigabitEthernet configured with a default gateway IP address. HSRP is used in a group of routers for selecting an active router and a standby router. In a group of router interfaces, the active router is the router of choice for routing packets; the standby router is the router that takes over when the active router fails or when preset conditions are met.

IPv6 hosts learn of available IPv6 routers through IPv6 neighbor discovery RA messages. These are multicast periodically, or may be solicited by hosts. HSRP is designed to provide only a virtual first hop for IPv6 hosts.

An HSRP IPv6 group has a virtual MAC address that is derived from the HSRP group number, and a virtual IPv6 link-local address that is, by default, derived from the HSRP virtual MAC address. Periodic RAs are sent for the HSRP virtual IPv6 link-local address when the HSRP group is active. These RAs stop after a final RA is sent when the group leaves the active state.

Periodic RAs for the interface link-local address stop after a final RA is sent while at least one virtual IPv6 link-local address is configured on the interface. No restrictions occur for the interface IPv6 link-local address other than that mentioned for the RAs. Other protocols continue to receive and send packets to this address.

HSRP uses a priority mechanism 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 IPv6 Virtual MAC Address Range

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

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

HSRP IPv6 UDP Port Number

Port number 2029 has been assigned to HSRP IPv6.

NSF and SSO-HSRP for IPv6 on VRF Interfaces

Cisco Nonstop Forwarding (NSF) and stateful switchover (SSO) are supported on HSRP for IPv6 on VRF interfaces, which makes HSRP for IPv6 VRF-aware.

For further information about SSO and NSF, see " Stateful Switchover " and " Cisco Nonstop Forwarding " in the *Cisco IOS XE High Availability Configuration Guide* .

How to Configure First Hop Redundancy Protocols in IPv6

Enabling an HSRP Group for IPv6 Operation

Prerequisites

HSRP version 2 must be enabled on an interface before HSRP IPv6 can be configured.

If an IPv6 address is entered, it must be link local. There are no HSRP IPv6 secondary addresses.

Enabling HSRP Version 2

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type number*
4. **standby version** {**1**| **2**

DETAILED STEPS

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

	Command or Action	Purpose
Step 3	**interface** *type number* **Example:** `Router(config)# interface GigabitEthernet 0/0/0`	Specifies an interface type and number, and places the router in interface configuration mode.
Step 4	**standby version** {**1**| **2** **Example:** `Router(config-if)# standby version 2`	Changes the version of the HSRP. • Version 1 is the default.

Enabling and Verifying an HSRP Group for IPv6 Operation

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

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

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

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

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ipv6 unicast-routing**
4. **interface** *type number*
5. **standby** [*group-number*] **ipv6** {*link-local-address* | **autoconfig**
6. **standby** [*group-number*] **preempt** [**delay minimum** *seconds* | **reload** *seconds* | **sync** *seconds*}]
7. **standby** [*group-number*] **priority** *priority*
8. **exit**
9. **show standby** [*type number* [*group*]] [**all** | **brief**
10. **show ipv6 interface** [**brief**] [*interface-type interface-number*] [**prefix**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**ipv6 unicast-routing** **Example:** `Router(config)# ipv6 unicast-routing`	Enables the forwarding of IPv6 unicast datagrams. • The **ipv6 unicast-routing** command must be enabled for HSRP for IPv6 to work.
Step 4	**interface** *type number* **Example:** `Router(config)# interface GigabitEthernet 0/0/0`	Specifies an interface type and number, and places the router in interface configuration mode.
Step 5	**standby** [*group-number*] **ipv6** {*link-local-address*	**autoconfig** **Example:** `Router(config-if)# standby 1 ipv6 autoconfig`	Activates the HSRP in IPv6.

	Command or Action	Purpose
Step 6	**standby** [*group-number*] **preempt** [**delay minimum** *seconds* | **reload** *seconds* | **sync** *seconds*}] **Example:** `Router(config-if)# standby 1 preempt`	Configures HSRP preemption and preemption delay.
Step 7	**standby** [*group-number*] **priority** *priority* **Example:** `Router(config-if)# standby 1 priority 110`	Configures HSRP priority.
Step 8	**exit** **Example:** `Router(config-if)# exit`	Returns the router to privileged EXEC mode.
Step 9	**show standby** [*type number* [*group*]] [**all** | **brief** **Example:** `Router# show standby`	Displays HSRP information.
Step 10	**show ipv6 interface** [**brief**] [*interface-type interface-number*] [**prefix** **Example:** `Router# show ipv6 interface GigabitEthernet 0/0/0`	Displays the usability status of interfaces configured for IPv6.

Configuration Examples for First Hop Redundancy Protocols in IPv6

Example: Enabling and Verifying an HSRP Group for IPv6 Operation

The following example shows configuration verification for an HSRP group for IPv6 that consists of Router1 and Router2. The **show standby** command is issued for each device to verify the device's configuration.

Router1 Configuration

```
interface GigabitEthernet0/0/0
description DATA VLAN for PCs
encapsulation dot1Q 100
ipv6 address 2001:DB8:CAFE:2100::BAD1:1010/64
standby version 2
standby 101 priority 120
standby 101 preempt delay minimum 30
standby 101 authentication ese
standby 101 track Serial0/1/0.17 90
standby 201 ipv6 autoconfig
standby 201 priority 120
standby 201 preempt delay minimum 30
standby 201 authentication ese
standby 201 track Serial0/1/0.17 90
Router1# show standby
GigabitFastEthernet0/0/0 - Group 101 (version 2)
State is Active
2 state changes, last state change 5w5d
Active virtual MAC address is 0000.0c9f.f065
Local virtual MAC address is 0000.0c9f.f065 (v2 default)
Hello time 3 sec, hold time 10 sec
Next hello sent in 2.296 secs
Authentication text "ese"
Preemption enabled, delay min 30 secs
Active router is local
Priority 120 (configured 120)
Track interface Serial0/1/0.17 state Up decrement 90
IP redundancy name is "hsrp-Fa0/0.100-101" (default)
GigabitEthernet0/0/0 - Group 201 (version 2)
State is Active
2 state changes, last state change 5w5d
Virtual IP address is FE80::5:73FF:FEA0:C9
Active virtual MAC address is 0005.73a0.00c9
Local virtual MAC address is 0005.73a0.00c9 (v2 IPv6 default)
Hello time 3 sec, hold time 10 sec
Next hello sent in 2.428 secs
Authentication text "ese"
Preemption enabled, delay min 30 secs
Active router is local
Standby router is FE80::20F:8FFF:FE37:3B70, priority 100 (expires in 7.856 sec)
Priority 120 (configured 120)
Track interface Serial0/1/0.17 state Up decrement 90
IP redundancy name is "hsrp-Fa0/0.100-201" (default)
```

Router2 Configuration

```
interface GigabitEthernet0/0/0
description DATA VLAN for Computers
encapsulation dot1Q 100
ipv6 address 2001:DB8:CAFE:2100::BAD1:1020/64
standby version 2
standby 101 preempt
standby 101 authentication ese
standby 201 ipv6 autoconfig
standby 201 preempt
standby 201 authentication ese
Router2# show standby
GigabitEthernet0/0/0 - Group 101 (version 2)
State is Standby
7 state changes, last state change 5w5d
Active virtual MAC address is 0000.0c9f.f065
Local virtual MAC address is 0000.0c9f.f065 (v2 default)
Hello time 3 sec, hold time 10 sec
Next hello sent in 0.936 secs
Authentication text "ese"
Preemption enabled
MAC address is 0012.7fc6.8f0c
Standby router is local
Priority 100 (default 100)
```

```
IP redundancy name is "hsrp-Fa0/0.100-101" (default)
GigabitEthernet0/0/0 - Group 201 (version 2)
State is Standby
7 state changes, last state change 5w5d
Virtual IP address is FE80::5:73FF:FEA0:C9
Active virtual MAC address is 0005.73a0.00c9
Local virtual MAC address is 0005.73a0.00c9 (v2 IPv6 default)
Hello time 3 sec, hold time 10 sec
Next hello sent in 0.936 secs
Authentication text "ese"
Preemption enabled
Active router is FE80::212:7FFF:FEC6:8F0C, priority 120 (expires in 7.548 sec)
MAC address is 0012.7fc6.8f0c
Standby router is local
Priority 100 (default 100)
IP redundancy name is "hsrp-Fa0/0.100-201" (default)
```

Additional References

Related Documents

Related Topic	Document Title
IPv6 link-local addresses and stateless autoconfiguration	" Implementing IPv6 Addressing and Basic Connectivity ," *Cisco IOS XE IPv6 Configuration Guide*
SSO and NSF	" Stateful Switchover " and " Cisco Nonstop Forwarding, " *Cisco IOS XE High Availability Configuration Guide* .
IPv6 supported feature list	" Start Here: Cisco IOS XE Software Release Specifics for IPv6 Features ," *Cisco IOS XE IPv6 Configuration Guide*
IPv6 commands: complete command syntax, command mode, command history, defaults, usage guidelines, and examples	*Cisco IOS IPv6 Command Reference*
Cisco IOS master command list, all releases	Cisco IOS Master Command List, All Releases

Standards

Standard	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
None	To locate and download MIBs for selected platforms, Cisco software releases, and feature sets, use Cisco MIB Locator found at the following URL: http://www.cisco.com/go/mibs

RFCs

RFC	Title
RFC 2281	*Cisco Hot Standby Router Protocol (HSRP)*

Technical Assistance

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

Feature Information for First Hop Redundancy Protocols in IPv6

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Table 11 Feature Information for First Hop Redundancy Protocols for IPv6

Feature Name	Releases	Feature Configuration Information
IPv6 services: HSRP for IPv6	Cisco IOS XE Release 3.1S Cisco IOS XE Release 3.3SG	The HSRP is an FHRP designed to allow for transparent failover of the first-hop IPv6 router. The following commands were modified for this release: **show ipv6 interface**, **show standby**, **standby ipv6**, **standby preempt**, **standby priority**, **standby version**.
NSF/SSO - HSRPv6 on VRF interfaces	Cisco IOS XE Release 3.1S	The NSF/SSO - HSRPv6 on VRF interfaces feature is supported in Cisco IOS XE Release 3.1S.
ISSU - HSRPv6 on VRF interfaces	Cisco IOS XE Release 3.1S	The ISSU - HSRPv6 on VRF interfaces feature is supported in Cisco IOS XE Release 3.1S.

Implementing IPsec in IPv6 Security

Cisco IOS IPv6 security features for your Cisco networking devices can protect your network against degradation or failure and also against data loss or compromise resulting from intentional attacks and from unintended but damaging mistakes by well-meaning network users.

Cisco IOS IPsec functionality provides network data encryption at the IP packet level, offering robust, standards-based security. IPsec provides data authentication and antireplay services in addition to data confidentiality services.

IPsec is a mandatory component of IPv6 specification. IPv6 IPsec tunnel mode and encapsulation is used to protect IPv6 unicast and multicast traffic. This document provides information about implementing IPsec in IPv6 security.

Finding Feature Information

Your software release may not support all the features documented in this module. For the latest caveats and feature information, see Bug Search Tool and the release notes for your platform and software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the feature information table at the end of this module.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Information About Implementing IPsec for IPv6 Security

IPsec for IPv6

IP Security, or IPsec, is a framework of open standards developed by the Internet Engineering Task Force (IETF) that provide security for transmission of sensitive information over unprotected networks such as

the Internet. IPsec acts at the network layer, protecting and authenticating IP packets between participating IPsec devices (peers), such as Cisco routers. IPsec provides the following optional network security services. In general, local security policy will dictate the use of one or more of these services:

- Data confidentiality--The IPsec sender can encrypt packets before sending them across a network.
- Data integrity--The IPsec receiver can authenticate packets sent by the IPsec sender to ensure that the data has not been altered during transmission.
- Data origin authentication--The IPsec receiver can authenticate the source of the IPsec packets sent. This service depends upon the data integrity service.
- Antireplay--The IPsec receiver can detect and reject replayed packets.

With IPsec, data can be sent across a public network without observation, modification, or spoofing. IPsec functionality is similar in both IPv6 and IPv4; however, site-to-site tunnel mode only is supported in IPv6.

In IPv6, IPsec is implemented using the AH authentication header and the ESP extension header. The authentication header provides integrity and authentication of the source. It also provides optional protection against replayed packets. The authentication header protects the integrity of most of the IP header fields and authenticates the source through a signature-based algorithm. The ESP header provides confidentiality, authentication of the source, connectionless integrity of the inner packet, antireplay, and limited traffic flow confidentiality.

The Internet Key Exchange (IKE) protocol is a key management protocol standard that is used in conjunction with IPsec. IPsec can be configured without IKE, but IKE enhances IPsec by providing additional features, flexibility, and ease of configuration for the IPsec standard.

IKE is a hybrid protocol that implements the Oakley key exchange and Skeme key exchange inside the Internet Security Association Key Management Protocol (ISAKMP) framework (ISAKMP, Oakley, and Skeme are security protocols implemented by IKE) (see the figure below). This functionality is similar to the security gateway model using IPv4 IPsec protection.

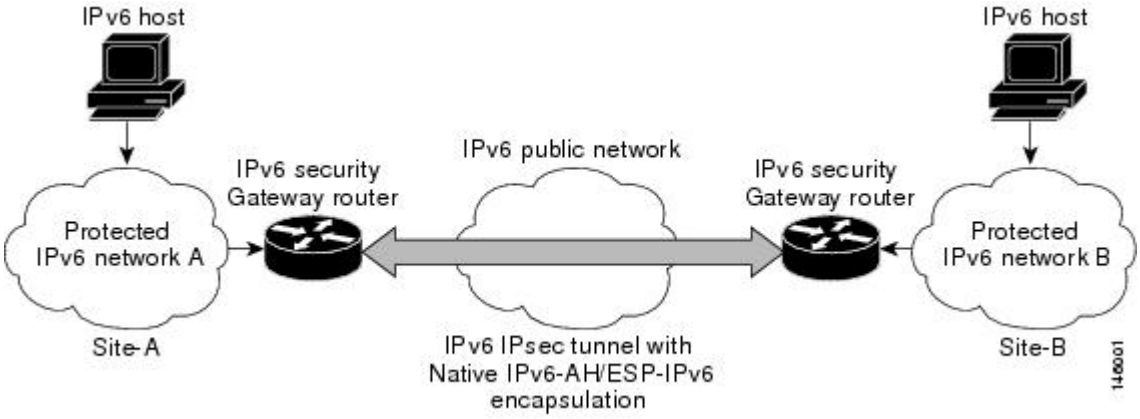
- IPv6 IPsec Site-to-Site Protection Using Virtual Tunnel Interface, page 190

IPv6 IPsec Site-to-Site Protection Using Virtual Tunnel Interface

The IPsec virtual tunnel interface (VTI) provides site-to-site IPv6 crypto protection of IPv6 traffic. Native IPv6 IPsec encapsulation is used to protect all types of IPv6 unicast and multicast traffic.

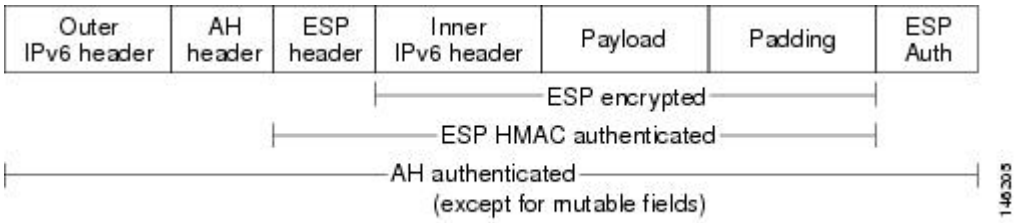
The IPsec VTI allows IPv6 routers to work as security gateways, establish IPsec tunnels between other security gateway routers, and provide crypto IPsec protection for traffic from internal networks when it is sent across the public IPv6 Internet (see the figure below). This functionality is similar to the security gateway model using IPv4 IPsec protection.

Figure 21 IPsec Tunnel Interface for IPv6



When the IPsec tunnel is configured, IKE and IPsec security associations (SAs) are negotiated and set up before the line protocol for the tunnel interface is changed to the UP state. The remote IKE peer is the same as the tunnel destination address; the local IKE peer will be the address picked from tunnel source interface which has the same IPv6 address scope as tunnel destination address.
The following figures shows the IPsec packet format.

Figure 22 IPv6 IPsec Packet Format



IPv6 over IPv4 GRE Tunnel Protection

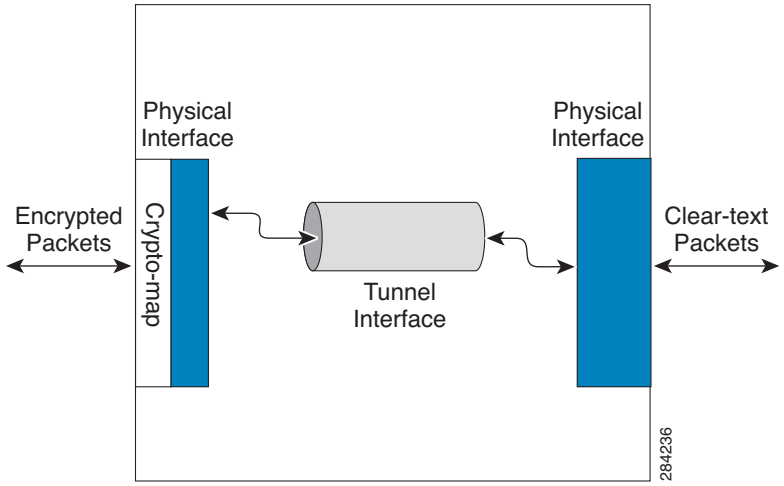
The IPv6 over IPv4 GRE tunnel protection feature allows both IPv6 unicast and multicast traffic to go through a protected GRE tunnel.

- GRE Tunnels with IPsec, page 191

GRE Tunnels with IPsec

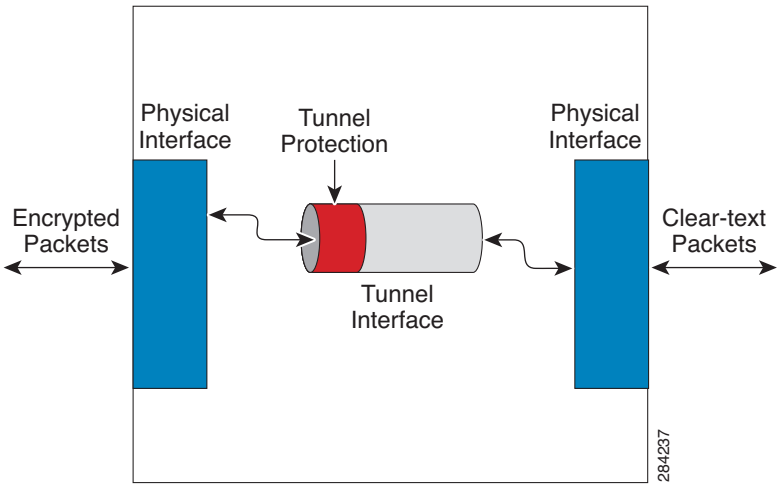
Generic routing encapsulation (GRE) tunnels sometimes are combined with IPSec, because IPSec does not support IPv6 multicast packets. This function prevents dynamic routing protocols from running successfully over an IPSec VPN network. Because GRE tunnels do support IPv6 multicast , a dynamic routing protocol can be run over a GRE tunnel. Once a dynamic routing protocol is configured over a GRE tunnel, you can encrypt the GRE IPv6 multicast packets using IPSec.
IPSec can encrypt GRE packets using a crypto map or tunnel protection. Both methods specify that IPSec encryption is performed after GRE encapsulation is configured. When a crypto map is used, encryption is applied to the outbound physical interfaces for the GRE tunnel packets. When tunnel protection is used, encryption is configured on the GRE tunnel interface.
The following figure shows encrypted packets that enter a router through a GRE tunnel interface using a crypto map on the physical interface. Once the packets are decrypted and decapsulated, they continue to their IP destination as clear text.

Figure 23 Using a Crypto Map to Configure IPv6 over IPv4 GRE Tunnel Encryption



The following figure shows encryption using tunnel protection command on the GRE tunnel interface. The encrypted packets enter the router through the tunnel interface and are decrypted and decapsulated before they continue to their destination as clear text.

Figure 24 Using Tunnel Protection to Configure IPv6 over IPv4 GRE Tunnel Encryption



There are two key differences in using the crypto map and tunnel protection methods:

- The IPSec crypto map is tied to the physical interface and is checked as packets are forwarded out through the physical interface. At this point, the GRE tunnel has already encapsulated the packet.
- Tunnel protection ties the encryption functionality to the GRE tunnel and is checked after the packet is GRE encapsulated but before the packet is handed to the physical interface.

How to Implement IPsec for IPv6 Security

Configuring a VTI for Site-to-Site IPv6 IPsec Protection

Defining an IKE Policy and a Preshared Key in IPv6

Because IKE negotiations must be protected, each IKE negotiation begins by agreement of both peers on a common (shared) IKE policy. This policy states which security parameters will be used to protect subsequent IKE negotiations and mandates how the peers are authenticated.

After the two peers agree upon a policy, the security parameters of the policy are identified by an SA established at each peer, and these SAs apply to all subsequent IKE traffic during the negotiation.

You can configure multiple, prioritized policies on each peer--each with a different combination of parameter values. However, at least one of these policies must contain exactly the same encryption, hash, authentication, and Diffie-Hellman parameter values as one of the policies on the remote peer. For each policy that you create, you assign a unique priority (1 through 10,000, with 1 being the highest priority).

Note If you are interoperating with a device that supports only one of the values for a parameter, your choice is limited to the value supported by the other device. Aside from this limitation, there is often a trade-off between security and performance, and many of these parameter values represent such a trade-off. You should evaluate the level of security risks for your network and your tolerance for these risks.

When the IKE negotiation begins, IKE searches for an IKE policy that is the same on both peers. The peer that initiates the negotiation will send all its policies to the remote peer, and the remote peer will try to find a match. The remote peer looks for a match by comparing its own highest priority policy against the policies received from the other peer. The remote peer checks each of its policies in order of its priority (highest priority first) until a match is found.

A match is made when both policies from the two peers contain the same encryption, hash, authentication, and Diffie-Hellman parameter values, and when the remote peer's policy specifies a lifetime that is less than or equal to the lifetime in the policy being compared. (If the lifetimes are not identical, the shorter lifetime--from the remote peer's policy--will be used.)

If a match is found, IKE will complete negotiation, and IPsec security associations will be created. If no acceptable match is found, IKE refuses negotiation and IPsec will not be established.

Note Depending on which authentication method is specified in a policy, additional configuration might be required. If a peer's policy does not have the required companion configuration, the peer will not submit the policy when attempting to find a matching policy with the remote peer.

You should set the ISAKMP identity for each peer that uses preshared keys in an IKE policy.

When two peers use IKE to establish IPsec SAs, each peer sends its identity to the remote peer. Each peer sends either its hostname or its IPv6 address, depending on how you have set the ISAKMP identity of the router.

By default, a peer's ISAKMP identity is the IPv6 address of the peer. If appropriate, you could change the identity to be the peer's hostname instead. As a general rule, set the identities of all peers the same way--either all peers should use their IPv6 addresses or all peers should use their hostnames. If some peers use their hostnames and some peers use their IPv6 addresses to identify themselves to each other, IKE negotiations could fail if the identity of a remote peer is not recognized and a DNS lookup is unable to resolve the identity.

Perform this task to create an IKE policy and a preshared key in IPv6.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **crypto isakmp policy** *priority*
4. **authentication** {**rsa-sig** | **rsa-encr** | **pre-share**}
5. **hash** {**sha** | **md5**}
6. **group** {**1** | **2** | **5**}
7. **encryption** {**des** | **3des** | **aes** | **aes 192** | **aes 256**}
8. **lifetime** *seconds*
9. **exit**
10. **crypto isakmp key** password-type keystring *keystring {* **address** *peer-address*| **ipv6** {*ipv6-address / ipv6-prefix*} | **hostname** *hostname} [* **no-xauth** *]*
11. **crypto keyring** *keyring-name* [**vrf** *fvrf-name*]
12. **pre-shared-key** {**address** *address* [*mask*] | **hostname** *hostname* | **ipv6** {*ipv6-address* | *ipv6-prefix*}} **key** *key*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**crypto isakmp policy** *priority* **Example:** `Router(config)# crypto isakmp policy 15`	Defines an IKE policy, and enters ISAKMP policy configuration mode. • Policy number 1 indicates the policy with the highest priority. The smaller the *priority* argument value, the higher the priority.
Step 4	**authentication** {**rsa-sig**	**rsa-encr**	**pre-share**} **Example:** `Router(config-isakmp-policy)# authentication pre-share`	Specifies the authentication method within an IKE policy. • The **rsa-sig** and **rsa-encr** keywords are not supported in IPv6.

	Command or Action	Purpose
Step 5	**hash** {**sha** | **md5**} **Example:** `Router(config-isakmp-policy)# hash md5`	Specifies the hash algorithm within an IKE policy.
Step 6	**group** {**1** | **2** | **5**} **Example:** `Router(config-isakmp-policy)# group 2`	Specifies the Diffie-Hellman group identifier within an IKE policy.
Step 7	**encryption** {**des** | **3des** | **aes** | **aes 192** | **aes 256**} **Example:** `Router(config-isakmp-policy)# encryption 3des`	Specifies the encryption algorithm within an IKE policy.
Step 8	**lifetime** *seconds* **Example:** `Router(config-isakmp-policy)# lifetime 43200`	Specifies the lifetime of an IKE SA. • Setting the IKE lifetime value is optional.
Step 9	**exit** **Example:** `Router(config-isakmp-policy)# exit`	Exits ISAKMP policy configuration mode and enter global configuration mode.
Step 10	**crypto isakmp key** password-type keystring *keystring {* **address** *peer-address*| **ipv6** {*ipv6-address / ipv6-prefix*} | **hostname** *hostname} [* **no-xauth** *]* **Example:** `Router(config)# crypto isakmp key 0 my-preshare-key-0 address ipv6 3ffe:1001::2/128`	Configures a preshared authentication key.
Step 11	**crypto keyring** *keyring-name* [**vrf** *fvrf-name*] **Example:** `Router(config)# crypto keyring keyring1`	Defines a crypto keyring to be used during IKE authentication and enters config-keyring mode.

	Command or Action	Purpose
Step 12	**pre-shared-key** {**address** *address* [*mask*] | **hostname** *hostname* | **ipv6** {*ipv6-address* | *ipv6-prefix*}} **key** *key* **Example:** `Router (config-keyring)# pre-shared-key ipv6 3FFE: 2002::A8BB:CCFF:FE01:2C02/128`	Defines a preshared key to be used for IKE authentication.

Configuring ISAKMP Aggressive Mode

You likely do not need to configure aggressive mode in a site-to-site scenario. The default mode is typically used.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **crypto isakmp peer** {**address** {*ipv4-address* | **ipv6** *ipv6-address ipv6-prefix-length*} | **hostname** *fqdn-hostname*}
4. **set aggressive-mode client-endpoint** {*client-endpoint* | **ipv6** *ipv6-address*}
5. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**crypto isakmp peer** {**address** {*ipv4-address* | **ipv6** *ipv6-address ipv6-prefix-length*} | **hostname** *fqdn-hostname*} **Example:** `Router(config)# crypto isakmp peer address ipv6 3FFE:2002::A8BB:CCFF:FE01:2C02/128`	Enables an IPsec peer for IKE querying for tunnel attributes.

	Command or Action	Purpose
Step 4	**set aggressive-mode client-endpoint** {*client-endpoint* | **ipv6** *ipv6-address*} **Example:** `Router(config-isakmp-peer)# set aggressive mode client-endpoint ipv6 3FFE:2002::A8BB:CCFF:FE01:2C02/128`	Defines the remote peer's IPv6 address, which will be used by aggressive mode negotiation. The remote peer's address is usually the client side's end-point address.
Step 5	**end** **Example:** `Router(config-isakmp-peer)# end`	Exits crypto ISAKMP peer configuration mode and returns to privileged EXEC mode.

Defining an IPsec Transform Set and IPsec Profile

Perform this task to define an IPsec transform set. A transform set is a combination of security protocols and algorithms that is acceptable to the IPsec routers.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **crypto ipsec transform-set** *transform-set-name transform1* [*transform2*] [*transform3*] [*transform4*]
4. **crypto ipsec profile** *name*
5. **set transform-set** *transform-set-name* [*transform-set-name2...transform-set-name6*

DETAILED STEPS

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

	Command or Action	Purpose
Step 3	**crypto ipsec transform-set** *transform-set-name transform1* [*transform2*] [*transform3*] [*transform4*] **Example:** `Router(config)# crypto ipsec transform-set myset0 ah-sha-hmac esp-3des`	Defines a transform set, and places the router in crypto transform configuration mode.
Step 4	**crypto ipsec profile** *name* **Example:** `Router(config)# crypto ipsec profile profile0`	Defines the IPsec parameters that are to be used for IPsec encryption between two IPsec routers.
Step 5	**set transform-set** *transform-set-name* [*transform-set-name2...transform-set-name6* **Example:** `Router (config-crypto-transform)# set-transform-set myset0`	Specifies which transform sets can be used with the crypto map entry.

Defining an ISAKMP Profile in IPv6

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **crypto isakmp profile** *profile-name* [**accounting** *aaalist*
4. **self-identity** {**address** | **address ipv6**] | **fqdn** | **user-fqdn** *user-fqdn*}
5. **match identity** {**group** *group-name* | **address** {*address* [*mask*] [*fvrf*] | **ipv6** *ipv6-address*} | **host** *host-name* | **host domain** *domain-name* | **user** *user-fqdn* | **user domain** *domain-name*}
6. **end**

DETAILED STEPS

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

	Command or Action	Purpose
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**crypto isakmp profile** *profile-name* [**accounting** *aaalist* **Example:** `Router(config)# crypto isakmp profile profile1`	Defines an ISAKMP profile and audits IPsec user sessions.
Step 4	**self-identity** {**address** | **address ipv6**] | **fqdn** | **user-fqdn** *user-fqdn*} **Example:** `Router(config-isakmp-profile)# self-identity address ipv6`	Defines the identity that the local IKE uses to identify itself to the remote peer.
Step 5	**match identity** {**group** *group-name* | **address** {*address* [*mask*] [*fvrf*] | **ipv6** *ipv6-address*} | **host** *host-name* | **host domain** *domain-name* | **user** *user-fqdn* | **user domain** *domain-name*} **Example:** `Router(config-isakmp-profile)# match identity address ipv6 3FFE: 2002::A8BB:CCFF:FE01:2C02/128`	Matches an identity from a remote peer in an ISAKMP profile.
Step 6	**end** **Example:** `Router(config-isakmp-profile)# end`	Exits ISAKMP profile configuration mode and returns to privileged EXEC mode.

Configuring IPv6 IPsec VTI

Use the **ipv6 unicast-routing** command to enable IPv6 unicast routing.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ipv6 unicast-routing**
4. **interface tunnel** *tunnel-number*
5. **ipv6 address** *ipv6-address/prefix*
6. **ipv6 enable**
7. **tunnel source** {*ip-address* | *ipv6-address* | *interface-type interface-number*}
8. **tunnel destination** {*host-name* | *ip-address* | *ipv6-address*}
9. **tunnel mode** {**aurp** | **cayman** | **dvmrp** | **eon** | **gre** | **gre multipoint** | **gre ipv6** | **ipip** [**decapsulate-any**] | **ipsec ipv4** | **iptalk** | **ipv6** | **ipsec ipv6** | **mpls** | **nos** | **rbscp**}
10. **tunnel protection ipsec profile** *name* [**shared**]
11. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**ipv6 unicast-routing** **Example:** `Router(config)# ipv6 unicast-routing`	Enables IPv6 unicast routing. You only need to enable IPv6 unicast routing once, not matter how many interface tunnels you want to configure.
Step 4	**interface tunnel** *tunnel-number* **Example:** `Router(config)# interface tunnel 0`	Specifies a tunnel interface and number, and enters interface configuration mode.

	Command or Action	Purpose
Step 5	**ipv6 address** *ipv6-address/prefix* **Example:** `Router(config-if)# ipv6 address 3FFE:C000:0:7::/64 eui-64`	Provides an IPv6 address to this tunnel interface, so that IPv6 traffic can be routed to this tunnel.
Step 6	**ipv6 enable** **Example:** `Router(config-if)# ipv6 enable`	Enables IPv6 on this tunnel interface.
Step 7	**tunnel source** {*ip-address* | *ipv6-address* | *interface-type interface-number*} **Example:** `Router(config-if)# tunnel source ethernet0`	Sets the source address for a tunnel interface.
Step 8	**tunnel destination** {*host-name* | *ip-address* | *ipv6-address*} **Example:** `Router(config-if)# tunnel destination 2001:DB8:1111:2222::1`	Specifies the destination for a tunnel interface.
Step 9	**tunnel mode** {**aurp** | **cayman** | **dvmrp** | **eon** | **gre** | **gre multipoint** | **gre ipv6** | **ipip** [**decapsulate-any**] | **ipsec ipv4** | **iptalk** | **ipv6** | **ipsec ipv6** | **mpls** | **nos** | **rbscp**} **Example:** `Router(config-if)# tunnel mode ipsec ipv6`	Sets the encapsulation mode for the tunnel interface. For IPsec, only the **ipsec ipv6** keywords are supported.
Step 10	**tunnel protection ipsec profile** *name* [**shared**] **Example:** `Router(config-if)# tunnel protection ipsec profile profile1`	Associates a tunnel interface with an IPsec profile. IPv6 does not support the **shared** keyword.
Step 11	**end** **Example:** `Router(config-if)# end`	Exits interface configuration mode and returns to privileged EXEC mode.

Verifying IPsec Tunnel Mode Configuration

SUMMARY STEPS

1. **show adjacency** [**summary** [*interface-type interface-number*]] | [**prefix**] [**interface** *interface-number*] [**connectionid** *id*] [**link** {**ipv4**| **ipv6** | **mpls**}] [**detail**]
2. **show crypto engine** {**accelerator** | **brief** | **configuration** | **connections** [**active** | **dh** | **dropped-packet** | **show**] | **qos**}
3. **show crypto ipsec sa** [**ipv6**] [*interface-type interface-number*] [**detailed**]
4. **show crypto isakmp peer** [**config** | **detail**]
5. **show crypto isakmp policy**
6. **show crypto isakmp profile** [**tag** *profilename* | **vrf** *vrfname*]
7. **show crypto map** [**interface** *interface* | **tag** *map-name*]
8. **show crypto session** [**detail**] | [**local** *ip-address* [**port** *local-port*] | [**remote** *ip-address* [**port** *remote-port*]] | **detail**] | **fvfr** *vrf-name* | **ivrf** *vrf-name*]
9. **show crypto socket**
10. **show ipv6 access-list** [*access-list-name*]
11. **show ipv6 cef** [*ipv6-prefix / prefix-length*] | [*interface-type interface-number*] [**longer-prefixes** | **similar-prefixes** | **detail** | **internal** | **platform** | **epoch** | **source**]]
12. **show interface** *type number* **stats**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**show adjacency** [**summary** [*interface-type interface-number*]] | [**prefix**] [**interface** *interface-number*] [**connectionid** *id*] [**link** {**ipv4**| **ipv6** | **mpls**}] [**detail**] **Example:** `Router# show adjacency detail`	Displays information about the Cisco Express Forwarding adjacency table or the hardware Layer 3-switching adjacency table.
Step 2	**show crypto engine** {**accelerator** | **brief** | **configuration** | **connections** [**active** | **dh** | **dropped-packet** | **show**] | **qos**} **Example:** `Router# show crypto engine connection active`	Displays a summary of the configuration information for the crypto engines.
Step 3	**show crypto ipsec sa** [**ipv6**] [*interface-type interface-number*] [**detailed**] **Example:** `Router# show crypto ipsec sa ipv6`	Displays the settings used by current SAs in IPv6.

	Command or Action	Purpose
Step 4	**show crypto isakmp peer** [**config** | **detail**] **Example:** `Router# show crypto isakmp peer detail`	Displays peer descriptions.
Step 5	**show crypto isakmp policy** **Example:** `Router# show crypto isakmp policy`	Displays the parameters for each IKE policy.
Step 6	**show crypto isakmp profile** [**tag** *profilename* | **vrf** *vrfname*] **Example:** `Router# show crypto isakmp profile`	Lists all the ISAKMP profiles that are defined on a router.
Step 7	**show crypto map** [**interface** *interface* | **tag** *map-name*] **Example:** `Router# show crypto map`	Displays the crypto map configuration. The crypto maps shown in this command output are dynamically generated. The user does not have to configure crypto maps.
Step 8	**show crypto session** [**detail**] | [**local** *ip-address* [**port** *local-port*] | [**remote** *ip-address* [**port** *remote-port*]] | **detail**] | **fvfr** *vrf-name* | **ivrf** *vrf-name*] **Example:** `Router# show crypto session`	Displays status information for active crypto sessions. IPv6 does not support the **fvfr** or **ivrf** keywords or the *vrf-name* argument.
Step 9	**show crypto socket** **Example:** `Router# show crypto socket`	Lists crypto sockets.
Step 10	**show ipv6 access-list** [*access-list-name*] **Example:** `Router# show ipv6 access-list`	Displays the contents of all current IPv6 access lists.

	Command or Action	Purpose
Step 11	**show ipv6 cef** [*ipv6-prefix / prefix-length*] | [*interface-type interface-number*] [**longer-prefixes** | **similar-prefixes** | **detail** | **internal** | **platform** | **epoch** | **source**]] **Example:** `Router# show ipv6 cef`	Displays entries in the IPv6 Forwarding Information Base (FIB).
Step 12	**show interface** *type number* **stats** **Example:** `Router# show interface fddi 3/0/0 stats`	Displays numbers of packets that were process switched, fast switched, and distributed switched.

Troubleshooting IPsec for IPv6 Configuration and Operation

SUMMARY STEPS

1. **enable**
2. **debug crypto ipsec**
3. **debug crypto engine packet** [**detail**]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router# enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**debug crypto ipsec** **Example:** `Router# debug crypto ipsec`	Displays IPsec network events.
Step 3	**debug crypto engine packet** [**detail**] **Example:** `Router# debug crypto engine packet`	Displays the contents of IPv6 packets. **Caution** Using this command could flood the system and increase CPU usage if several packets are being encrypted.

• Examples, page 205

Examples

This section provides the following output examples:

Sample Output from the show crypto ipsec sa Command

The following is sample output from the **show crypto ipsec sa**command:

```
Router# show crypto ipsec sa
interface: Tunnel0
     Crypto map tag: Tunnel0-head-0, local addr 3FFE:2002::A8BB:CCFF:FE01:9002
    protected vrf: (none)
    local  ident (addr/mask/prot/port): (::/0/0/0)
    remote ident (addr/mask/prot/port): (::/0/0/0)
    current_peer 3FFE:2002::A8BB:CCFF:FE01:2C02 port 500
      PERMIT, flags={origin_is_acl,}
     #pkts encaps: 133, #pkts encrypt: 133, #pkts digest: 133
     #pkts decaps: 133, #pkts decrypt: 133, #pkts verify: 133
     #pkts compressed: 0, #pkts decompressed: 0
     #pkts not compressed: 0, #pkts compr. failed: 0
     #pkts not decompressed: 0, #pkts decompress failed: 0
     #send errors 60, #recv errors 0
      local crypto endpt.: 3FFE:2002::A8BB:CCFF:FE01:9002,
      remote crypto endpt.: 3FFE:2002::A8BB:CCFF:FE01:2C02
      path mtu 1514, ip mtu 1514
      current outbound spi: 0x28551D9A(676666778)
      inbound esp sas:
       spi: 0x2104850C(553944332)
         transform: esp-des ,
         in use settings ={Tunnel, }
         conn id: 93, flow_id: SW:93, crypto map: Tunnel0-head-0
         sa timing: remaining key lifetime (k/sec): (4397507/148)
         IV size: 8 bytes
         replay detection support: Y
         Status: ACTIVE
      inbound ah sas:
       spi: 0x967698CB(2524354763)
         transform: ah-sha-hmac ,
         in use settings ={Tunnel, }
         conn id: 93, flow_id: SW:93, crypto map: Tunnel0-head-0
         sa timing: remaining key lifetime (k/sec): (4397507/147)
         replay detection support: Y
         Status: ACTIVE
      inbound pcp sas:
      outbound esp sas:
       spi: 0x28551D9A(676666778)
         transform: esp-des ,
         in use settings ={Tunnel, }
         conn id: 94, flow_id: SW:94, crypto map: Tunnel0-head-0
         sa timing: remaining key lifetime (k/sec): (4397508/147)
         IV size: 8 bytes
         replay detection support: Y
         Status: ACTIVE
      outbound ah sas:
       spi: 0xA83E05B5(2822636981)
         transform: ah-sha-hmac ,
         in use settings ={Tunnel, }
         conn id: 94, flow_id: SW:94, crypto map: Tunnel0-head-0
         sa timing: remaining key lifetime (k/sec): (4397508/147)
         replay detection support: Y
         Status: ACTIVE
      outbound pcp sas:
```

Sample Output from the show crypto isakmp peer Command

The following sample output shows peer descriptions on an IPv6 router:

```
Router# show crypto isakmp peer detail
```

```
Peer: 2001:DB8:0:1::1 Port: 500 Local: 2001:DB8:0:2::1
Phase1 id: 2001:DB8:0:1::1
flags:
NAS Port: 0 (Normal)
IKE SAs: 1 IPsec SA bundles: 1
last_locker: 0x141A188, last_last_locker: 0x0
last_unlocker: 0x0, last_last_unlocker: 0x0
```

Sample Output from the show crypto isakmp profile Command

The following sample output shows the ISAKMP profiles that are defined on an IPv6 router:

```
Router# show crypto isakmp profile
ISAKMP PROFILE tom
Identities matched are:
ipv6-address 2001:DB8:0:1::1/32
Certificate maps matched are:
Identity presented is: ipv6-address fqdn
keyring(s): <none>
trustpoint(s): <all>
```

Sample Output from the show crypto isakmp sa Command

The following sample output shows the SAs of an active IPv6 device. The IPv4 device is inactive.

```
Router# show crypto isakmp sa detail

Codes: C - IKE configuration mode, D - Dead Peer Detection
       K - Keepalives, N - NAT-traversal
       X - IKE Extended Authentication
       psk - Preshared key, rsig - RSA signature
       renc - RSA encryption
IPv4 Crypto ISAKMP SA

C-id  Local           Remote          I-VRF    Status Encr Hash Auth DH
Lifetime Cap.

IPv6 Crypto ISAKMP SA

  dst: 3FFE:2002::A8BB:CCFF:FE01:2C02
  src: 3FFE:2002::A8BB:CCFF:FE01:9002
  conn-id: 1001  I-VRF:          Status: ACTIVE Encr: des  Hash: sha  Auth:
psk
  DH: 1  Lifetime: 23:45:00 Cap: D    Engine-id:Conn-id = SW:1

  dst: 3FFE:2002::A8BB:CCFF:FE01:2C02
  src: 3FFE:2002::A8BB:CCFF:FE01:9002
  conn-id: 1002  I-VRF:          Status: ACTIVE Encr: des  Hash: sha  Auth: psk
  DH: 1  Lifetime: 23:45:01 Cap: D    Engine-id:Conn-id = SW:2
```

Sample Output from the show crypto map Command

The following sample output shows the dynamically generated crypto maps of an active IPv6 device:

```
Router# show crypto map

Crypto Map "Tunnel1-head-0" 65536 ipsec-isakmp
        Profile name: profile0
        Security association lifetime: 4608000 kilobytes/300 seconds
        PFS (Y/N): N
```

```
        Transform sets={
                ts,
        }
Crypto Map "Tunnel1-head-0" 65537
        Map is a PROFILE INSTANCE.
        Peer = 2001:1::2
IPv6 access list Tunnel1-head-0-ACL (crypto)
    permit ipv6 any any (61445999 matches) sequence 1
        Current peer: 2001:1::2
        Security association lifetime: 4608000 kilobytes/300 seconds
        PFS (Y/N): N
        Transform sets={
          ts,
        }
        Interfaces using crypto map Tunnel1-head-0:
        Tunnel1
```

Sample Output from the show crypto session Command

The following output from the show crypto session command provides details on currently active crypto sessions:

```
Router# show crypto session detail
Crypto session current status
Code: C - IKE Configuration mode, D - Dead Peer Detection K - Keepalives, N - NAT-
traversal, X - IKE Extended Authentication
Interface: Tunnel1
Session status: UP-ACTIVE
Peer: 2001:1::1 port 500 fvrf: (none) ivrf: (none)
       Phase1_id: 2001:1::1
       Desc: (none)
   IKE SA: local 2001:1::2/500
           remote 2001:1::1/500 Active
           Capabilities:(none) connid:14001 lifetime:00:04:32
   IPSEC FLOW: permit ipv6 ::/0 ::/0
         Active SAs: 4, origin: crypto map
         Inbound:  #pkts dec'ed 42641 drop 0 life (KB/Sec) 4534375/72
         Outbound: #pkts enc'ed 6734980 drop 0 life (KB/Sec) 2392402/72
```

Configuration Examples for IPsec for IPv6 Security

Example: Configuring a VTI for Site-to-Site IPv6 IPsec Protection

```
crypto isakmp policy 1
  authentication pre-share
!
crypto isakmp key myPreshareKey0 address ipv6 3FFE:2002::A8BB:CCFF:FE01:2C02/128
crypto isakmp keepalive 30 30
!
crypto ipsec transform-set 3des ah-sha-hmac esp-3des
!
crypto ipsec profile profile0
  set transform-set 3des
!
ipv6 cef
!
interface Tunnel0
  ipv6 address 3FFE:1001::/64 eui-64
  ipv6 enable
  ipv6 cef
  tunnel source Ethernet2/0
  tunnel destination 3FFE:2002::A8BB:CCFF:FE01:2C02
```

```
tunnel mode ipsec ipv6
tunnel protection ipsec profile profile0
```

Additional References

Related Documents

Related Topic	Document Title
OSPFv3 authentication support with IPsec	Implementing OSPFv3
IPsec VTI information	IPsec Virtual Tunnel Interface
IPv6 supported feature list	Start Here: Cisco IOS Software Release Specifics for IPv6 Features
IPv6 commands: complete command syntax, command mode, defaults, usage guidelines, and examples	*Cisco IOS IPv6 Command Reference*
IPv4 security configuration tasks	*Cisco IOS Security Configuration Guide*
IPv4 security commands: complete command syntax, command mode, defaults, usage guidelines, and examples	*Cisco IOS Security Command Reference*

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
None	To locate and download MIBs for selected platforms, Cisco software releases, and feature sets, use Cisco MIB Locator found at the following URL: http://www.cisco.com/go/mibs

RFCs

RFCs	Title
RFC 2401	*Security Architecture for the Internet Protocol*

RFCs	Title
RFC 2402	*IP Authentication Header*
RFC 2404	*The Use of Hash Message Authentication Code Federal Information Processing Standard 180-1 within Encapsulating Security Payload and Authentication Header*
RFC 2406	*IP Encapsulating Security Payload (ESP)*
RFC 2407	*The Internet Security Domain of Interpretation for ISAKMP*
RFC 2408	*Internet Security Association and Key Management Protocol (ISAKMP)*
RFC 2409	*Internet Key Exchange (IKE)*
RFC 2460	*Internet Protocol, Version 6 (IPv6) Specification*
RFC 2474	*Definition of the Differentiated Services Field (DS Field) in the IPv4 and IPv6 Headers*
RFC 3576	*Change of Authorization*
RFC 4109	*Algorithms for Internet Key Exchange version 1 (IKEv1)*
RFC 4302	*IP Authentication Header*
RFC 4306	*Internet Key Exchange (IKEv2) Protocol*
RFC 4308	*Cryptographic Suites for IPsec*

Technical Assistance

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

Feature Information for Implementing IPsec in IPv6 Security

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software

release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Table 12 Feature Information for Implementing IPsec in IPv6 Security

Feature Name	Releases	Feature Information
IPv6 IPsec VPN	Cisco IOS XE Release 2.4	The following commands were introduced or modified: **authentication (IKE policy), crypto ipsec profile, crypto isakmp identity, crypto isakmp key, crypto isakmp peer, crypto isakmp policy, crypto isakmp profile, crypto keyring, debug crypto ipv6 ipsec, debug crypto ipv6 packet, deny (IPv6), encryption (IKE policy), group (IKE policy), hash (IKE policy), lifetime (IKE policy), match identity, permit (IPv6), pre-shared-key, self-identity, set aggressive-mode client-endpoint, set transform-set, show crypto engine, show crypto ipsec policy, show crypto ipsec sa, show crypto isakmp key, show crypto isakmp peers, show crypto isakmp policy, show crypto isakmp profile, show crypto map (IPsec), show crypto session, show crypto socket**
IPSec Virtual Tunnel Interface	Cisco IOS XE Release 2.4	
IPv6 over v4 GRE Tunnel Protection	Cisco IOS XE Release 3.5S	

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.

Implementing IS-IS for IPv6

This module describes how to configure Integrated Intermediate System-to-Intermediate System (IS-IS) for IPv6. IS-IS is an Interior Gateway Protocol (IGP) that advertises link-state information throughout the network to create a picture of the network topology. IS-IS is an Open Systems Interconnection (OSI) hierarchical routing protocol that designates an intermediate system as a Level 1 or Level 2 device. Level 2 devices route between Level 1 areas to create an intradomain routing backbone. Integrated IS-IS uses a single routing algorithm to support several network address families, such as IPv6, IPv4, and OSI.

Finding Feature Information

Your software release may not support all the features documented in this module. For the latest caveats and feature information, see Bug Search Tool and the release notes for your platform and software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the feature information table at the end of this module.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Information About Implementing IS-IS for IPv6

IS-IS Enhancements for IPv6

IS-IS in IPv6 functions the same and offers many of the same benefits as IS-IS in IPv4. IPv6 enhancements to IS-IS allow IS-IS to advertise IPv6 prefixes in addition to IPv4 and OSI routes. Extensions to the IS-IS command-line interface (CLI) allow configuration of IPv6-specific parameters. IPv6 IS-IS extends the address families supported by IS-IS to include IPv6, in addition to OSI and IPv4.

IS-IS in IPv6 supports either single-topology mode or multiple topology mode.

IS-IS Single-Topology Support for IPv6

Single-topology support for IPv6 allows IS-IS for IPv6 to be configured on interfaces along with other network protocols (for example, IPv4 and Connectionless Network Service [CLNS]). All interfaces must be configured with the identical set of network address families. In addition, all routers in the IS-IS area (for Level 1 routing) or the domain (for Level 2 routing) must support the identical set of network layer address families on all interfaces.

When single-topology support for IPv6 is being used, either old- or new-style TLVs may be used. However, the TLVs used to advertise reachability to IPv6 prefixes use extended metrics. Cisco routers do not allow an interface metric to be set to a value greater than 63 if the configuration is not set to support only new-style TLVs for IPv4. In single-topology IPv6 mode, the configured metric is always the same for both IPv4 and IPv6.

IS-IS Multitopology Support for IPv6

IS-IS multitopology support for IPv6 allows IS-IS to maintain a set of independent topologies within a single area or domain. This mode removes the restriction that all interfaces on which IS-IS is configured must support the identical set of network address families. It also removes the restriction that all routers in the IS-IS area (for Level 1 routing) or domain (for Level 2 routing) must support the identical set of network layer address families. Because multiple SPFs are performed, one for each configured topology, it is sufficient that connectivity exists among a subset of the routers in the area or domain for a given network address family to be routable.

You can use the **isis ipv6 metric** command to configure different metrics on an interface for IPv6 and IPv4.

When multitopology support for IPv6 is used, use the **metric-style wide**command to configure IS-IS to use new-style TLVs because TLVs used to advertise IPv6 information in link-state packets (LSPs) are defined to use only extended metrics.

Transition from Single-Topology to Multitopology Support for IPv6

All routers in the area or domain must use the same type of IPv6 support, either single-topology or multitopology. A router operating in multitopology mode will not recognize the ability of the single-topology mode router to support IPv6 traffic, which will lead to holes in the IPv6 topology. To transition from single-topology support to the more flexible multitopology support, a multitopology transition mode is provided.

The multitopology transition mode allows a network operating in single-topology IS-IS IPv6 support mode to continue to work while upgrading routers to include multitopology IS-IS IPv6 support. While in transition mode, both types of TLVs (single-topology and multitopology) are sent in LSPs for all configured IPv6 addresses, but the router continues to operate in single-topology mode (that is, the topological restrictions of the single-topology mode are still in effect). After all routers in the area or domain have been upgraded to support multitopology IPv6 and are operating in transition mode, transition mode can be removed from the configuration. Once all routers in the area or domain are operating in multitopology IPv6 mode, the topological restrictions of single-topology mode are no longer in effect.

IPv6 IS-IS Local RIB

A router that is running IS-IS IPv6 maintains a local RIB in which it stores all routes to destinations it has learned from its neighbors. At the end of each SPF, IS-IS attempts to install the best (that is, the least-cost) routes to a destination present in the local RIB in the global IPv6 routing table.

How to Implement IS-IS for IPv6

When configuring supported routing protocols in IPv6, you must create the routing process, enable the routing process on interfaces, and customize the routing protocol for your particular network.

Configuring Single-Topology IS-IS for IPv6

Configuring IS-IS comprises two activities. The first activity creates an IS-IS routing process and is performed using protocol-independent IS-IS commands. The second activity in configuring IPv6 IS-IS configures the operation of the IS-IS protocol on an interface.

Before configuring the router to run IPv6 IS-IS, globally enable IPv6 using the **ipv6 unicast-routing** global configuration command.

Note If you are using IS-IS single-topology support for IPv6, IPv4, or both IPv6 and IPv4, you may configure both IPv6 and IPv4 on an IS-IS interface for Level 1, Level 2, or both Level 1 and Level 2. However, if both IPv6 and IPv4 are configured on the same interface, they must be running the same IS-IS level. That is, IPv4 cannot be configured to run on IS-IS Level 1 only on a specified GigabitEthernet or FastEthernet interface while IPv6 is configured to run IS-IS Level 2 only on the same GigabitEthernet or FastEthernet interface.

>

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **router isis** *area-tag*
4. **net** *network-entity-title*
5. **exit**
6. **interface** *type number*
7. **ipv6 address** {*ipv6-address* / *prefix-length* | *prefix-name sub-bits*/*prefix-length*
8. **ipv6 router isis** *area-name*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**router isis** *area-tag* **Example:** `Router(config)# router isis area2`	Enables IS-IS for the specified IS-IS routing process, and enters router configuration mode.
Step 4	**net** *network-entity-title* **Example:** `Router(config-router)# net 49.0001.0000.0000.000c.00`	Configures an IS-IS network entity title (NET) for the routing process. • The *network-entity-title* argument defines the area addresses for the IS-IS area and the system ID of the router.
Step 5	**exit** **Example:** `Router(config-router)# exit`	Exits router configuration mode and enters global configuration mode.

	Command or Action	Purpose
Step 6	**interface** *type number* **Example:** `Router(config)# interface GigabitEthernet 0/0/1`	Specifies the interface type and number, and enters interface configuration mode.
Step 7	**ipv6 address** {*ipv6-address* / *prefix-length* | *prefix-name sub-bits*/*prefix-length* **Example:** `Router(config-if)# ipv6 address 2001:DB8::3/64`	Specifies the IPv6 network assigned to the interface and enables IPv6 processing on the interface. **Note** Refer to the Implementing IPv6 Addressing and Basic Connectivity module for more information on configuring IPv6 addresses.
Step 8	**ipv6 router isis** *area-name* **Example:** `Router(config-if)# ipv6 router isis area2`	Enables the specified IPv6 IS-IS routing process on an interface.

Configuring Multitopology IS-IS for IPv6

When multitopology IS-IS for IPv6 is configured, the **transition** keyword allows a user who is working with the single-topology SPF mode of IS-IS IPv6 to continue to work while upgrading to multitopology IS-IS. After every router is configured with the **transition** keyword, users can remove the **transition** keyword on each router. When transition mode is not enabled, IPv6 connectivity between routers operating in single-topology mode and routers operating in multitopology mode is not possible.

You can continue to use the existing IPv6 topology while upgrading to multitopology IS-IS. The optional **isis ipv6 metric** command allows you to differentiate between link costs for IPv6 and IPv4 traffic when operating in multitopology mode.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **router isis** *area-tag*
4. **metric-style wide** [**transition**] [**level-1** | **level-2** | **level-1-2**
5. **address-family ipv6** [**unicast** | **multicast**]
6. **multi-topology** [**transition**]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**router isis** *area-tag* **Example:** `Router(config)# router isis area2`	Enables IS-IS for the specified IS-IS routing process, and enters router configuration mode.
Step 4	**metric-style wide** [**transition**] [**level-1** | **level-2** | **level-1-2** **Example:** `Router(config-router)# metric-style wide level-1`	Configures a router running IS-IS to generate and accept only new-style TLVs.
Step 5	**address-family ipv6** [**unicast** | **multicast**] **Example:** `Router(config-router)# address-family ipv6`	Specifies the IPv6 address family, and enters address family configuration mode. • The **unicast** keyword specifies the unicast IPv6 unicast address family. By default, the router is placed in configuration mode for the unicast IPv6 address family if the **unicast** keyword is not specified with the **address-family ipv6** command.
Step 6	**multi-topology** [**transition**] **Example:** `Router(config-router-af)# multi-topology`	Enables multitopology IS-IS for IPv6. • The optional **transition** keyword allows an IS-IS IPv6 user to continue to use single-topology mode while upgrading to multitopology mode.

Customizing IPv6 IS-IS

Perform this task to configure a new administrative distance for IPv6 IS-IS, configure the maximum number of equal-cost paths that IPv6 IS-IS will support, configure summary prefixes for IPv6 IS-IS, and configure an IS-IS instance to advertise the default IPv6 route (::/0). It also explains how to configure the

hold-down period between partial route calculations (PRCs) and how often Cisco IOS XE software performs the SPF calculation when using multitopology IS-IS.

You can customize IS-IS multitopology for IPv6 for your network, but you likely will not need to do so. The defaults for this feature are set to meet the requirements of most customers and features. If you change the defaults, refer to the IPv4 configuration guide and the IPv6 command reference to find the appropriate syntax.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **router isis** *area-tag*
4. **address-family ipv6** [**unicast** | **multicast**]
5. **default-information originate** [**route-map** *map-name*]
6. **distance** *value*
7. **maximum-paths** *number-paths*
8. **summary-prefix** *ipv6-prefix prefix-length* **level-1** | **level-1-2**| **level-2**]
9. **prc-interval** *seconds* [*initial-wait*] [*secondary-wait*
10. **spf-interval** [**level-1** | **level-2**] *seconds initial-wait*] [*secondary-wait*
11. **exit**
12. **interface** *type number*
13. **isis ipv6 metric** *metric-value* [**level-1** | **level-2** | **level-1-2**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**router isis** *area-tag* **Example:** `Router(config)# router isis area2`	Enables IS-IS for the specified IS-IS routing process, and enters router configuration mode.

	Command or Action	Purpose
Step 4	**address-family ipv6** [**unicast** | **multicast**] **Example:** `Router(config-router)# address-family ipv6`	Specifies the IPv6 address family, and enters address family configuration mode. • The **unicast** keyword specifies the unicast IPv6 unicast address family. By default, the router is placed in configuration mode for the unicast IPv6 address family if the **unicast** keyword is not specified with the **address-family ipv6** command.
Step 5	**default-information originate** [**route-map** *map-name*] **Example:** `Router(config-router-af)# default-information originate`	(Optional) Injects a default IPv6 route into an IS-IS routing domain. • The **route-map** keyword and *map-name* argument specify the conditions under which the IPv6 default route is advertised. • If the **route map** keyword is omitted, then the IPv6 default route will be unconditionally advertised at Level 2.
Step 6	**distance** *value* **Example:** `Router(config-router-af)# distance 90`	(Optional) Defines an administrative distance for IPv6 IS-IS routes in the IPv6 routing table. • The *value*argument is an integer from 10 to 254. (The values 0 to 9 are reserved for internal use).
Step 7	**maximum-paths** *number-paths* **Example:** `Router(config-router-af)# maximum-paths 3`	(Optional) Defines the maximum number of equal-cost routes that IPv6 IS-IS can support. • This command also supports IPv6 Border Gateway Protocol (BGP) and Routing Information Protocol (RIP). • The *number-paths* argument is an integer from 1 to 64. The default for BGP is one path; the default for IS-IS and RIP is 16 paths.
Step 8	**summary-prefix** *ipv6-prefix prefix-length* **level-1** | **level-1-2**| **level-2**] **Example:** `Router(config-router-af)# summary-prefix 2001:DB8::/24`	(Optional) Allows a Level 1-2 router to summarize Level 1 prefixes at Level 2, instead of advertising the Level 1 prefixes directly when the router advertises the summary. • The *ipv6-prefix* argument in the **summary-prefix** command must be in the form documented in RFC 2373 where the address is specified in hexadecimal using 16-bit values between colons. • The *prefix-length* argument 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 mark must precede the decimal value.

	Command or Action	Purpose
Step 9	**prc-interval** *seconds* [*initial-wait*] [*secondary-wait* **Example:** `Router(config-router-af)# prc-interval 20`	(Optional) Configures the hold-down period between PRCs for multitopology IS-IS for IPv6.
Step 10	**spf-interval** [**level-1** | **level-2**] *seconds initial-wait*] [*secondary-wait* **Example:** `Router(config-router-af)# spf-interval 30`	(Optional) Configures how often Cisco IOS XE software performs the SPF calculation for multitopology IS-IS for IPv6.
Step 11	**exit** **Example:** `Router(config-router-af)# exit`	Exits address family configuration mode, and returns the router to router configuration mode. • Repeat this step to exit router configuration mode and return the router to global configuration mode.
Step 12	**interface** *type number* **Example:** `Router(config-router)# interface GigabitEthernet 0/0/1`	Specifies the interface type and number, and enters interface configuration mode.
Step 13	**isis ipv6 metric** *metric-value* [**level-1** | **level-2** | **level-1-2** **Example:** `Router(config-if)# isis ipv6 metric 20`	(Optional) Configures the value of an multitopology IS-IS for IPv6 metric.

Redistributing Routes into an IPv6 IS-IS Routing Process

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **router isis** *area-tag*
4. **address-family ipv6** [**unicast** | **multicast**]
5. **redistribute** *source-protocol process-id*] [**include-connected**] [*target-protocol-options*] [*source-protocol-options*]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**router isis** *area-tag* **Example:** `Router(config)# router isis area2`	Enables IS-IS for the specified IS-IS routing process, and enters router configuration mode.
Step 4	**address-family ipv6** [**unicast**	**multicast**] **Example:** `Router(config-router)# address-family ipv6`	Specifies the IPv6 address family, and enters address family configuration mode. • The **unicast** keyword specifies the unicast IPv6 unicast address family. By default, the router is placed in configuration mode for the unicast IPv6 address family if the **unicast** keyword is not specified with the **address-family ipv6** command.

	Command or Action	Purpose
Step 5	**redistribute** *source-protocol process-id*] [**include-connected**] [*target-protocol-options*] [*source-protocol-options*] **Example:** `Router(config-router-af)# redistribute bgp 64500 metric 100 route-map isismap`	Redistributes routes from the specified protocol into the IS-IS process. • The *source-protocol* argument can be one of the following keywords: **bgp**, **connected**, **isis**, **rip**, or **static**. • Only the arguments and keywords relevant to this task are specified here.

Redistributing IPv6 IS-IS Routes Between IS-IS Levels

Perform this task to redistribute IPv6 routes learned at one IS-IS level into a different level.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **router isis** *area-tag*
4. **address-family ipv6** [**unicast** | **multicast**]
5. **redistribute isis** [*process-id*] {**level-1** | **level-2**} **into** {**level-1** | **level-2**} **distribute-list** *list-name*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**router isis** *area-tag* **Example:** `Router(config)# router isis area2`	Enables IS-IS for the specified IS-IS routing process, and enters router configuration mode.

	Command or Action	Purpose
Step 4	**address-family ipv6** [**unicast** | **multicast**] **Example:** `Router(config-router)# address-family ipv6`	Specifies the IPv6 address family, and enters address family configuration mode. • The **unicast** keyword specifies the unicast IPv6 unicast address family. By default, the router is placed in configuration mode for the unicast IPv6 address family if the **unicast** keyword is not specified with the **address-family ipv6** command.
Step 5	**redistribute isis** [*process-id*] {**level-1** | **level-2**} **into** {**level-1** | **level-2**} **distribute-list** *list-name* **Example:** `Router(config-router-af)# redistribute isis level-1 into level-2`	Redistributes IPv6 routes from one IS-IS level into another IS-IS level. • By default, the routes learned by Level 1 instances are redistributed by the Level 2 instance. **Note** The *protocol* argument must be **isis** in this configuration of the **redistribute** command. Only the arguments and keywords relevant to this task are specified here.

Disabling IPv6 Protocol-Support Consistency Checks

Perform this task to disable protocol-support consistency checks in IPv6 single-topology mode.

For single-topology IS-IS IPv6, routers must be configured to run the same set of address families. IS-IS performs consistency checks on hello packets and will reject hello packets that do not have the same set of configured address families. For example, a router running IS-IS for both IPv4 and IPv6 will not form an adjacency with a router running IS-IS for IPv4 or IPv6 only. In order to allow adjacency to be formed in mismatched address-families network, the **adjacency-check** command in IPv6 address family configuration mode must be disabled.

Note Entering the **no adjacency-check** command can adversely affect your network configuration. Enter the **no adjacency-check** command only when you are running IPv4 IS-IS on all your routers and you want to add IPv6 IS-IS to your network but you need to maintain all your adjacencies during the transition. When the IPv6 IS-IS configuration is complete, remove the **no adjacency-check**command from the configuration.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **router isis** *area-tag*
4. **address-family ipv6** [**unicast** | **multicast**]
5. **no adjacency-check**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**router isis** *area-tag* **Example:** `Router(config)# router isis area2`	Enables IS-IS for the specified IS-IS routing process, and enters router configuration mode.
Step 4	**address-family ipv6** [**unicast** | **multicast**] **Example:** `Router(config-router)# address-family ipv6`	Specifies the IPv6 address family, and enters address family configuration mode. • The **unicast** keyword specifies the unicast IPv6 unicast address family. By default, the router is placed in configuration mode for the unicast IPv6 address family if the **unicast** keyword is not specified with the **address-family ipv6** command.
Step 5	**no adjacency-check** **Example:** `Router(config-router-af)# no adjacency-check`	Disables the IPv6 protocol-support consistency checks performed on hello packets, allowing IPv6 to be introduced into an IPv4-only network without disrupting existing adjacencies. • The **adjacency-check**command is enabled by default.

Disabling IPv4 Subnet Consistency Checks

Perform this task to disable IPv4 subnet consistency checking when forming adjacencies. Cisco IOS XE software historically makes checks on hello packets to ensure that the IPv4 address is present and has a consistent subnet with the neighbor from which the hello packets are received. To disable this check, use the **no adjacency-check** command in the router configuration mode. However, if multitopology IS-IS is configured, this check is automatically suppressed, because multitopology IS-IS requires routers to form an adjacency regardless of whether or not all routers on a LAN support a common protocol.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **router isis** *area-tag*
4. **no adjacency-check**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**router isis** *area-tag* **Example:** `Router(config)# router isis area2`	Enables IS-IS for the specified IS-IS routing process, and enters router configuration mode.
Step 4	**no adjacency-check** **Example:** `Router(config-router-af)# no adjacency-check`	Disables the IPv6 protocol-support consistency checks performed on hello packets, allowing IPv6 to be introduced into an IPv4-only network without disrupting existing adjacencies. • The **adjacency-check**command is enabled by default.

Verifying IPv6 IS-IS Configuration and Operation

SUMMARY STEPS

1. **enable**
2. **show ipv6 protocols** [**summary**]
3. **show isis** [*process-tag*] [**ipv6** | *****] **topology**
4. **show clns** [*process-tag*] **neighbors** *interface-type interface-number*] [**area**] [**detail**]
5. **show clns** *area-tag* **is-neighbors** [*type number*] [**detail**]
6. **show isis** [*process-tag*] **database** [**level-1**] [**level-2**] [**l1**] [**l2**] [**detail**] [**lspid**]
7. **show isis ipv6 rib** [*ipv6-prefix*]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**show ipv6 protocols** [**summary**] **Example:** `Router# show ipv6 protocols`	Displays the parameters and current state of the active IPv6 routing processes.
Step 3	**show isis** [*process-tag*] [**ipv6** | *****] **topology** **Example:** `Router# show isis topology`	Displays a list of all connected routers running IS-IS in all areas.
Step 4	**show clns** [*process-tag*] **neighbors** *interface-type interface-number*] [**area**] [**detail**] **Example:** `Router# show clns neighbors detail`	Displays end system (ES), intermediate system (IS), and multitopology IS-IS (M-ISIS) neighbors.
Step 5	**show clns** *area-tag* **is-neighbors** [*type number*] [**detail**] **Example:** `Router# show clns is-neighbors detail`	Displays IS-IS adjacency information for IS-IS neighbors. • Use the **detail** keyword to display the IPv6 link-local addresses of the neighbors.
Step 6	**show isis** [*process-tag*] **database** [**level-1**] [**level-2**] [**l1**] [**l2**] [**detail**] [**lspid**] **Example:** `Router# show isis database detail`	Displays the IS-IS link-state database. • In this example, the contents of each LSP are displayed using the **detail** keyword.
Step 7	**show isis ipv6 rib** [*ipv6-prefix*] **Example:** `Router# show isis ipv6 rib`	Displays the IPv6 local RIB.

Examples

This section provides the following output examples:

Sample Output for the show ipv6 protocols Command

In the following example, output information about the parameters and current state of that active IPv6 routing processes is displayed using the **show ipv6 protocols**command:

```
Router# show ipv6 protocols
IPv6 Routing Protocol is "connected"
IPv6 Routing Protocol is "static"
IPv6 Routing Protocol is "isis"
  Interfaces:
    GigabitEthernet0/0/3
    GigabitEthernet0/0/1
    Serial1/0/1
    Loopback1 (Passive)
    Loopback2 (Passive)
    Loopback3 (Passive)
    Loopback4 (Passive)
    Loopback5 (Passive)
  Redistribution:
    Redistributing protocol static at level 1
  Address Summarization:
    L2: 2001:DB8:33::/16  advertised with metric 0
    L2: 2001:DB8:44::/16  advertised with metric 20
    L2: 2001:DB8:66::/16  advertised with metric 10
    L2: 2001:DB8:77::/16  advertised with metric 10
```

Sample Output for the show isis topology Command

In the following example, output information about all connected routers running IS-IS in all areas is displayed using the **show isis topology**command:

```
Router# show isis topology
IS-IS paths to level-1 routers
System Id       Metric  Next-Hop        Interface       SNPA
0000.0000.000C
0000.0000.000D  20      0000.0000.00AA  Se1/0/1         *HDLC*
0000.0000.000F  10      0000.0000.000F  GE0/0/1         0050.e2e5.d01d
0000.0000.00AA  10      0000.0000.00AA  Se1/0/1         *HDLC*
IS-IS paths to level-2 routers
System Id       Metric  Next-Hop        Interface       SNPA
0000.0000.000A  10      0000.0000.000A  GE0/0/3         0010.f68d.f063
0000.0000.000B  20      0000.0000.000A  GE0/0/3         0010.f68d.f063
0000.0000.000C  --
0000.0000.000D  30      0000.0000.000A  GE0/0/3         0010.f68d.f063
0000.0000.000E  30      0000.0000.000A  GE0/0/3         0010.f68d.f063
```

Sample Output for the show clns neighbors Command

In the following example, detailed output information that displays both end system (ES) and intermediate system (IS) neighbors is displayed using the **show clns neighbors** command with the **detail** keyword.

```
Router# show clns neighbors detail
System Id          Interface    SNPA            State  Holdtime  Type Protocol
0000.0000.0007     GE3/3        aa00.0400.6408  UP     26        L1   IS-IS
Area Address(es): 20
IP Address(es): 172.16.0.42*
Uptime: 00:21:49
0000.0C00.0C35     GE3/2        0000.0c00.0c36  Up     91        L1   IS-IS
Area Address(es): 20
IP Address(es): 192.168.0.42*
Uptime: 00:21:52
0800.2B16.24EA     GE3/3        aa00.0400.2d05  Up     27        L1   M-ISIS
Area Address(es): 20
IP Address(es): 192.168.0.42*
IPv6 Address(es): FE80::2B0:8EFF:FE31:EC57
Uptime: 00:00:27
0800.2B14.060E     GE3/2        aa00.0400.9205  Up     8         L1   IS-IS
Area Address(es): 20
IP Address(es): 192.168.0.30*
Uptime: 00:21:52
```

Sample Output for the show clns is-neighbors Command

In the following example, output information to confirm that the local router has formed all the necessary IS-IS adjacencies with other IS-IS neighbors is displayed using the **show clns is-neighbors**command. To display the IPv6 link-local addresses of the neighbors, specify the **detail** keyword.

```
Router# show clns is-neighbors detail
System Id      Interface   State  Type Priority  Circuit Id         Format
0000.0000.00AA Se1/0/1     Up     L1   0         00                 Phase V
  Area Address(es): 49.0001
  IPv6 Address(es): FE80::YYYY:D37C:C854:5
  Uptime: 17:21:38
0000.0000.000F Et0/0/1     Up     L1   64        0000.0000.000C.02  Phase V
  Area Address(es): 49.0001
  IPv6 Address(es): FE80::XXXX:E2FF:FEE5:D01D
  Uptime: 17:21:41
0000.0000.000A Et0/0/3     Up     L2   64        0000.0000.000C.01  Phase V
  Area Address(es): 49.000b
  IPv6 Address(es): FE80::ZZZZ:F6FF:FE8D:F063
  Uptime: 17:22:06
```

Sample Output for the show isis database Command

In the following example, detailed output information about LSPs received from other routers and the IPv6 prefixes they are advertising is displayed using the **show isis database**command with the **detail** keyword specified:

```
Router# show isis database detail
IS-IS Level-1 Link State Database
LSPID                 LSP Seq Num  LSP Checksum  LSP Holdtime  ATT/P/OL
0000.0C00.0C35.00-00  0x0000000C   0x5696        325           0/0/0
  Area Address: 47.0004.004D.0001
  Area Address: 39.0001
  Metric: 10   IS 0000.0C00.62E6.03
  Metric: 0    ES 0000.0C00.0C35
 --More--
0000.0C00.40AF.00-00* 0x00000009   0x8452        608           1/0/0
  Area Address: 47.0004.004D.0001
  Topology: IPv4 (0x0) IPv6 (0x2)
  NLPID: 0xCC 0x8E
  IP Address: 172.16.21.49
  Metric: 10   IS 0800.2B16.24EA.01
  Metric: 10   IS 0000.0C00.62E6.03
```

```
  Metric: 0    ES 0000.0C00.40AF
  IPv6 Address: 2001:DB8::/32
  Metric: 10   IPv6 (MT-IPv6) 2001:DB8::/64
  Metric: 5    IS-Extended cisco.03
  Metric: 10   IS-Extended cisco1.03
  Metric: 10    IS (MT-IPv6) cisco.03
IS-IS Level-2 Link State Database:
LSPID                 LSP Seq Num  LSP Checksum  LSP Holdtime      ATT/P/OL
0000.0000.000A.00-00  0x00000059   0x378A        949               0/0/0
  Area Address: 49.000b
  NLPID:        0x8E
  IPv6 Address: 2001:DB8:1:1:1:1:1:1
  Metric: 10         IPv6 2001:DB8:2:YYYY::/64
  Metric: 10         IPv6 2001:DB8:3:YYYY::/64
  Metric: 10         IPv6 2001:DB8:2:YYYY::/64
  Metric: 10         IS-Extended 0000.0000.000A.01
  Metric: 10         IS-Extended 0000.0000.000B.00
  Metric: 10         IS-Extended 0000.0000.000C.01
  Metric: 0          IPv6 11:1:YYYY:1:1:1:1:1/128
  Metric: 0          IPv6 11:2:YYYY:1:1:1:1:1/128
  Metric: 0          IPv6 11:3:YYYY:1:1:1:1:1/128
  Metric: 0          IPv6 11:4:YYYY:1:1:1:1:1/128
  Metric: 0          IPv6 11:5:YYYY:1:1:1:1:1/128
0000.0000.000A.01-00  0x00000050   0xB0AF        491               0/0/0
  Metric: 0          IS-Extended 0000.0000.000A.00
  Metric: 0          IS-Extended 0000.0000.000B.00
```

Sample Output for the show isis ipv6 rib Command

The following example shows output from the **show isis ipv6 rib** command. An asterisk (*) indicates prefixes that have been installed in the master IPv6 RIB as IS-IS routes. Following each prefix is a list of all paths in order of preference, with optimal paths listed first and suboptimal paths listed after optimal paths.

```
Router# show isis ipv6 rib

IS-IS IPv6 process "", local RIB
  2001:DB8:88:1::/64
    via FE80::210:7BFF:FEC2:ACC9/GigabitEthernet2/0/0, type L2  metric 20 LSP [3/7]
    via FE80::210:7BFF:FEC2:ACCC/GigabitEthernet2/1/0, type L2  metric 20 LSP [3/7]
* 2001:DB8:1357:1::/64
    via FE80::202:7DFF:FE1A:9471/GigabitEthernet2/1/0, type L2  metric 10 LSP [4/9]
* 2001:DB8:45A::/64
    via FE80::210:7BFF:FEC2:ACC9/GigabitEthernet2/0/0, type L1  metric 20 LSP [C/6]
    via FE80::210:7BFF:FEC2:ACCC/GigabitEthernet2/1/0, type L1  metric 20 LSP [C/6]
    via FE80::210:7BFF:FEC2:ACC9/GigabitEthernet2/0/0, type L2  metric 20 LSP [3/7]
    via FE80::210:7BFF:FEC2:ACCC/GigabitEthernet2/1/0, type L2  metric 20 LSP [3/7]
```

Configuration Examples for IPv6 IS-IS

Example Configuring Single-Topology IS-IS for IPv6

The following example enables single-topology mode, creates an IS-IS process, defines the NET, configures an IPv6 address on an interface, and configures the interface to run IPv6 IS-IS:

```
ipv6 unicast-routing
!
router isis
 net 49.0001.0000.0000.000c.00
 exit
interface GigabitEthernet0/0/1
 ipv6 address 2001:DB8::3/64
 ipv6 router isis area2
```

Example: Customizing IPv6 IS-IS

The following example advertises the IPv6 default route (::/0)--with an origin of GigabitEthernet interface 0/0/1--with all other routes in router updates sent on GigabitEthernet interface 0/0/1. This example also sets an administrative distance for IPv6 IS-IS to 90, defines the maximum number of equal-cost paths that IPv6 IS-IS will support as 3, and configures a summary prefix of 2001:DB8::/24 for IPv6 IS-IS.

```
router isis
 address-family ipv6
 default-information originate
 distance 90
 maximum-paths 3
 summary-prefix 2001:DB8::/24
 exit
```

Example: Redistributing Routes into an IPv6 IS-IS Routing Process

The following example redistributes IPv6 BGP routes into the IPv6 IS-IS Level 2 routing process:

```
router isis
 address-family ipv6
 redistribute bgp 64500 metric 100 route-map isismap
 exit
```

Example: Redistributing IPv6 IS-IS Routes Between IS-IS Levels

The following example redistributes IPv6 IS-IS Level 1 routes into the IPv6 IS-IS Level 2 routing process:

```
router isis
 address-family ipv6
 redistribute isis level-1 into level-2
```

Example: Disabling IPv6 Protocol-Support Consistency Checks

The following example disables the **adjacency-check** command to allow a network administrator to configure IPv6 IS-IS on the router without disrupting the existing adjacencies:

```
router isis
 address-family ipv6
 no adjacency-check
```

Example Configuring Multitopology IS-IS for IPv6

The following example configures multitopology IS-IS in IPv6 after you have configured IS-IS for IPv6:

```
router isis
 metric-style wide
 address-family ipv6
 multi-topology
```

Example Configuring the IS-IS IPv6 Metric for Multitopology IS-IS

The following example sets the value of an IS-IS IPv6 metric to 20:

```
interface GigabitEthernet 0/0/1
 isis ipv6 metric 20
```

Additional References

Related Documents

Related Topic	Document Title
IPv6 supported feature list	Start Here: Cisco IOS XE Software Release Specifics for IPv6 Features , *Cisco IOS XE IPv6 Configuration Guide*
IPv6 commands: complete command syntax, command mode, command history, defaults, usage guidelines, and examples	*Cisco IOS IPv6 Command Reference*
IS-IS commands: complete command syntax, command mode, defaults, usage guidelines, and examples	*Cisco IOS IP Routing Protocols Command Reference*
Cisco IOS master command list, all releases	Cisco IOS Master Command List, All Releases

MIBs

MIBs	MIBs Link
• CISCO-IETF-IP-FORWARD-MIB • CISCO-IETF-IP-MIB	To locate and download MIBs for selected platforms, Cisco software releases, and feature sets, use Cisco MIB Locator found at the following URL: http://www.cisco.com/go/mibs

RFCs

RFCs	Title
RFC 1195	*Use of OSI IS-IS for Routing in TCP/IP and Dual Environments*
RFC 5120	*M-ISIS: Multi-Topology (MT) Routing in IS-IS* , October 2, 2002
RFC 5308	*Routing IPv6 with IS-IS* , October 31, 2002

Technical Assistance

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

Feature Information for Implementing IS-IS for IPv6

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Table 13 Feature Information for Implementing IS-IS for IPv6

Feature Name	Releases	Feature Information
IPv6 Routing--IS-IS Local RIB	Cisco IOS XE Release 2.6	A router that is running IS-IS IPv6 maintains a local RIB in which it stores all routes to destinations it has learned from its neighbors. The following commands were modified by this feature: **show isis ipv6 rib**

Feature Name	Releases	Feature Information
IPv6 Routing--IS-IS Multitopology Support for IPv6	Cisco IOS XE Release 2.6	IS-IS multitopology support for IPv6 allows IS-IS to maintain a set of independent topologies within a single area or domain. The following commands were modified by this feature: **address-family ipv6 (IS-IS)**, **debug isis spf-events**, **isis ipv6 metric**, **multi-topology**, **prc-interval (IPv6)**, **show clns neighbors**, **spf-interval (IPv6)**
IPv6 Routing--IS-IS Support for IPv6	Cisco IOS XE Release 2.4	IPv6 enhancements to IS-IS allow IS-IS to advertise IPv6 prefixes in addition to IPv4 and OSI routes. The following commands were modified by this feature: **address-family ipv6 (IS-IS)**. **adjacency-check**, **default-information originate (IPv6 IS-IS)**. **distance (IPv6)**. **ipv6 router isis**, **show clns neighbors**, **show ipv6 protocols**, **show isis database**, **show isis topology**, **summary-prefix (IPv6 IS-IS)**
IPv6 Routing--Route Redistribution	Cisco IOS XE Release 2.4	IS-IS for IPv6 supports redistributing routes into an IPv6 IS-IS routing process and redistributing IPv6 IS-IS routes between IS-IS levels. The following commands were modified by this feature: **address-family ipv6 (IS-IS)**, **redistribute isis (IPv6)**

Implementing IPv6 for Network Management

This document describes the concepts and commands used to manage Cisco applications over IPv6 and to implement IPv6 for network management.

Finding Feature Information

Your software release may not support all the features documented in this module. For the latest caveats and feature information, see Bug Search Tool and the release notes for your platform and software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the feature information table at the end of this module.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Information About Implementing IPv6 for Network Management

Telnet Access over IPv6

The Telnet client and server in Cisco software support IPv6 connections. A user can establish a Telnet session directly to the device using an IPv6 Telnet client, or an IPv6 Telnet connection can be initiated

from the device. A vty interface and password must be created in order to enable Telnet access to an IPv6 device.

TFTP IPv6 Support

TFTP is designed to transfer files over the network from one host to another using the most minimal set of functionality possible. TFTP uses a client/server model in which clients can request to copy files to or from a server. TFTP uses UDP over IPv4 or IPv6 as its transport, and it can work over IPv4 and IPv6 network layers.

- TFTP File Downloading for IPv6, page 238

TFTP File Downloading for IPv6

IPv6 supports TFTP file downloading and uploading using the **copy** command. The **copy** command accepts a destination IPv6 address or IPv6 hostname as an argument and saves the running configuration of the device to an IPv6 TFTP server, as follows:

```
Device# copy running-config tftp://[3ffe:xxxx:c18:1:290:27ff:fe3a:9e9a]/running-config
```

ping and traceroute Commands in IPv6

The **ping** command accepts a destination IPv6 address or IPv6 hostname as an argument and sends Internet Control Message Protocol version 6 (ICMPv6) echo request messages to the specified destination. The ICMPv6 echo reply messages are reported on the console. Extended ping functionality is also supported in IPv6.

The **traceroute** command accepts a destination IPv6 address or IPv6 hostname as an argument and will generate IPv6 traffic to report each IPv6 hop used to reach the destination address.

SSH over an IPv6 Transport

Secure shell (SSH) SSH in IPv6 functions the same and offers the same benefits as SSH in IPv4. The SSH server feature enables an SSH client to make a secure, encrypted connection to a Cisco device, and the SSH client feature enables a Cisco device to make a secure, encrypted connection to another Cisco device or to any other device running an SSH server. IPv6 enhancements to SSH consist of support for IPv6 addresses that enable a Cisco device to accept and establish secure, encrypted connections with remote IPv6 nodes over an IPv6 transport.

SNMP over an IPv6 Transport

Simple Network Management Protocol (SNMP) can be configured over IPv6 transport so that an IPv6 host can perform SNMP queries and receive SNMP notifications from a device running IPv6 software. The SNMP agent and related MIBs have been enhanced to support IPv6 addressing. This feature uses the data encryption standard (3DES) and advanced encryption standard (AES) message encryption.

- Cisco IOS XE IPv6 MIBs, page 238
- MIBs Supported for IPv6, page 239

Cisco IOS XE IPv6 MIBs

Cisco has long supported IP-MIB and IP-FORWARD-MIB in IPv4. CISCO-IETF-IP-MIB and CISCO-IETF-IP-FORWARDING-MIB are IPv6 MIBs that are defined as being protocol-independent, but are

implemented only for IPv6 objects and tables. IP-MIB and IP-FORWARD-MIB were updated to RFC 4293 and RFC 4292 standards, as follows:

- The upgrade is backward-compatible; all IP-MIB and IP-FORWARD-MIB objects and tables still appear.
- IP-MIB and IP-FORWARD-MIB include new IPv6-only, IPv4-only, and protocol-version independent (PVI) objects and tables. However, IPv6 supports IPv6-only and the new IPv6 part of the PVI objects and tables in these MIBs.

MIBs Supported for IPv6

The following MIBs are supported for IPv6:

- CISCO-CONFIG-COPY-MIB
- CISCO-CONFIG-MAN-MIB
- CISCO-DATA-COLLECTION-MIB
- CISCO-FLASH-MIB
- CISCO-SNMP-TARGET-EXT-MIB
- ENTITY-MIB
- IP-FORWARD-MIB
- IP-MIB
- NOTIFICATION-LOG-MIB
- SNMP-TARGET-MIB

CISCO-CONFIG-COPY-MIB and CISCO-FLASH-MIB support IPv6 addressing when TFTP, remote copy protocol (rcp), or FTP is used.

Cisco IOS XE IPv6 Embedded Management Components

This section describes Cisco IOS XE software embedded management components that have IPv6-compliant operability in IPv6 and dual-stack IPv6 and IPv4 networks.

Syslog

The Cisco system message logging (syslog) process in IPv6 allows users to log syslog messages to external syslog servers and hosts with IPv6 addresses. This implementation allows user to specify an IPv4-based logging host (syslog server) by providing the host's IP address in IPv4 format (for example, 192.168.0.0) or IPv6 format (for example, 2001:DB8:A00:1::1/64).

TCL

Tool command language (TCL) is used in Cisco software for IPv6 to support features such as embedded syslog manager (ESM), embedded event manager (EEM), interactive voice response (IVR), and tclsh parser mode. TCL supports both initiating (client) and listening (server) sockets.

CNS Agents

IPv6 addressing is supported in the Cisco Networking Services (CNS) subsystem. CNS is a foundation technology for linking users to networking services, and it provides the infrastructure for the automated configuration of large numbers of network devices. Many IPv6 networks are complex, with many devices, and each device must be configured individually. When standard configurations do not exist or have been modified, the time involved in initial installation and subsequent upgrading is considerable. ISPs need a method for sending out partial configurations to introduce new services.

To address all these issues, CNS was designed to provide "plug-and-play" network services using a central directory service and distributed agents. CNS features include CNS agents and a flow-through provisioning structure. CNS flow-through provisioning uses the CNS configuration and event agents to provide an automated workflow, eliminating the need for an onsite technician.

IPv6 addressing supports the CNS agents described in the following sections:

- CNS Configuration Agent, page 240
- CNS Event Agent, page 240
- CNS EXEC Agent, page 240
- CNS Image Agent, page 240

CNS Configuration Agent

The CNS configuration agent is involved in the initial configuration and subsequent partial configurations on a Cisco device. The configuration agent uses a CNS configuration engine to provide methods for automating initial Cisco device configurations, incremental configurations, and synchronized configuration updates, and the configuration engine reports the status of the configuration load as an event to which a network monitoring or workflow application can subscribe.

CNS Event Agent

The CNS event agent provides a transport connection to the CNS event bus for all other CNS agents. No event can be sent to the device by the configuration engine until the CNS event agent is operational and has successfully built a connection between the configuration engine and the device.

The event agent uses a CNS configuration engine to provide methods for automating initial Cisco device configurations, incremental configurations, and synchronized configuration updates.

CNS EXEC Agent

The CNS EXEC agent allows a remote application to execute a CLI command in EXEC mode on a Cisco device by sending an event message that contains the command.

CNS Image Agent

Administrators maintaining large networks of Cisco devices need an automated mechanism to load image files onto large numbers of remote devices. Network management applications are useful to determine which images to run and how to manage images received from the Cisco online software center. Other image distribution solutions do not scale to cover thousands of devices and cannot distribute images to devices behind a firewall or using Network Address Translation (NAT). The CNS image agent enables the managed device to initiate a network connection and request an image download allowing devices using NAT, or behind firewalls, to access the image server.

The CNS image agent can be configured to use the CNS event bus. To use the CNS event bus, the CNS event agent must be enabled and connected to the CNS event gateway in the CNS Configuration Engine.

The CNS image agent can also use an HTTP server that understands the CNS image agent protocol. Deployment of CNS image agent operations can use both the CNS event bus and an HTTP server.

Config Logger

Config logger tracks and reports configuration changes. Config logger supports two content types:

- Plain text--With plain-text format, the config logger reports configuration changes only.
- XML--The config logger uses XML to report the configuration change details (for example, what changed, who changed it, when changes were made, parser return code [PRC] values, and incremental NVGEN results).

IP SLAs for IPv6

Cisco IP Service Level Agreements (SLAs) are a portfolio of technology embedded in most devices that run Cisco software that allows Cisco customers to analyze IPv6 service levels for IPv6 applications and services, increase productivity, lower operational costs, and reduce the frequency of network outages. IP SLAs uses active traffic monitoring--the generation of traffic in a continuous, reliable, and predictable manner--for measuring network performance.

The following Cisco IP SLAs are supported for IPv6:

- Internet Control Message Protocol (ICMP) echo operation--Used to monitor end-to-end response time between a Cisco device and other devices using IPv4 or IPv6. ICMP echo is useful for troubleshooting network connectivity issues.
- TCP connect operation--Used to measure the response time taken to perform a TCP Connect operation between a Cisco device and other devices using IPv4 or IPv6.
- User Datagram Protocol (UDP) echo operation--Used to monitor end-to-end response time between a Cisco router and devices using IPv4 or IPv6 .
- UDP jitter operation--Used to analyze round-trip delay, one-way delay, one-way jitter, one-way packet loss, and connectivity in networks that carry UDP traffic in IPv4 or IPv6 networks.
- UDP jitter operation--Used to monitor VoIP quality levels in your network, allowing you to guarantee VoIP quality levels to your users in IPv4 or IPv6 networks.

How to Implement IPv6 for Network Management

- Enabling Telnet Access to an IPv6 Device and Establishing a Telnet Session, page 242
- Enabling SSH on an IPv6 Router, page 243
- Configuring an SNMP Notification Server over IPv6, page 245
- Configuring Cisco IOS XE IPv6 Embedded Management Components, page 248

Enabling Telnet Access to an IPv6 Device and Establishing a Telnet Session

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ipv6 host** *name* [*port*] *ipv6-address*
4. **line** [**aux** | **console** | **tty** | **vty**] *line-number* [*ending-line-number*]
5. **password** *password*
6. **login** [**local** | **tacacs**]
7. **ipv6 access-class** *ipv6-access-list-name* {**in** | **out**]
8. **telnet** *host* [*port*] [*keyword*]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**ipv6 host** *name* [*port*] *ipv6-address* **Example:** `Device(config)# ipv6 host cisco-sj 2001:DB8:20:1::12`	Defines a static hostname-to-address mapping in the hostname cache.
Step 4	**line** [**aux**	**console**	**tty**	**vty**] *line-number* [*ending-line-number*] **Example:** `Device(config)# line vty 0 4`	Creates a vty interface.

	Command or Action	Purpose
Step 5	**password** *password* **Example:** `Device(config)# password hostword`	Creates a password that enables Telnet.
Step 6	**login** [**local** | **tacacs**] **Example:** `Device(config)# login tacacs`	(Optional) Enables password checking at login.
Step 7	**ipv6 access-class** *ipv6-access-list-name* {**in** | **out**] **Example:** `Device(config)# ipv6 access-list hostlist`	(Optional) Adds an IPv6 access list to the line interface. • Using this command restricts remote access to sessions that match the access list.
Step 8	**telnet** *host* [*port*] [*keyword*] **Example:** `Device(config)# telnet cisco-sj`	Establishes a Telnet session from a device to a remote host using either the hostname or the IPv6 address. • The Telnet session can be established to a device name or to an IPv6 address.

Enabling SSH on an IPv6 Router

If you do not configure SSH parameters, then the default values will be used.

Prior to configuring SSH over an IPv6 transport, ensure that the following conditions exist:

- An IPsec (Data Encryption Standard [DES] or 3DES) encryption software image is loaded on your router. IPv6 transport for the SSH server and SSH client requires an IPsec encryption software image.
- A hostname and host domain are configured for your router. Refer to the "Mapping Hostnames to IPv6 Addresses" section of Implementing IPv6 Addressing and Basic Connectivity for information on assigning hostnames to IPv6 addresses and specifying default domain names that can be used by both IPv4 and IPv6.
- A Rivest, Shamir, and Adelman (RSA) key pair, which automatically enables SSH, is generated for your router. RSA is the public key cryptographic system developed by Ron Rivest, Adi Shamir, and Leonard Adelman. RSA keys come in pairs: one public key and one private key.
- A user authentication mechanism for local or remote access is configured on your router.

Note The basic restrictions for SSH over an IPv4 transport listed in the " Configuring Secure Shell " chapter of *Cisco IOS XE Security Configuration Guide* apply to SSH over an IPv6 transport. In addition to the restrictions listed in that chapter, the use of locally stored usernames and passwords is the only user authentication mechanism supported by SSH over an IPv6 transport; the TACACS+ and RADIUS user authentication mechanisms are not supported over an IPv6 transport.

Note To authenticate SSH clients, configure TACACS+ or RADIUS over an IPv4 transport and then an SSH server over an IPv6 transport.

>

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ip ssh** [**timeout** *seconds* | **authentication-retries** *integer*]
4. **exit**
5. **ssh** [**-v** {**1** | **2**}] [**-c** {**3des** | **aes128-cbc** | **aes192-cbc** | **aes256-cbc**}] [**-l** *userid* | **-l** *userid* **:** *{number} {ip-address}* | **-l** *userid* **:rotary** *{number} {ip-address}*] [**-m** {**hmac-md5** | **hmac-md5-96** | **hmac-sha1** | **hmac-sha1-96**}] [**-o numberofpasswordprompts***n*] [**-p** *port-num*] {*ip-addr* | *hostname*} [*command*

DETAILED STEPS

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

	Command or Action	Purpose
Step 3	**ip ssh** [**timeout** *seconds* | **authentication-retries** *integer*] **Example:** `Router(config)# ip ssh timeout 100 authentication-retries 2`	Configures SSH control variables on your router. • You can specify the timeout in seconds, not to exceed 120 seconds. The default is 120. This setting applies to the SSH negotiation phase. Once the EXEC session starts, the standard timeouts configured for the vty apply. By default, five vty lines are defined (0-4); therefore, five terminal sessions are possible. After the SSH executes a shell, the vty timeout starts. The vty timeout defaults to 10 minutes. • You can also specify the number of authentication retries, not to exceed five authentication retries. The default is three.
Step 4	**exit** **Example:** `Router(config)# exit`	Exits configuration mode, and returns the router to privileged EXEC mode.
Step 5	**ssh** [**-v** {**1** | **2**}] [**-c** {**3des** | **aes128-cbc** | **aes192-cbc** | **aes256-cbc**}] [**-l** *userid* | **-l** *userid* **:** *{number} {ip-address}* | **-l** *userid* **:rotary** *{number} {ip-address}*] [**-m** {**hmac-md5** | **hmac-md5-96** | **hmac-sha1** | **hmac-sha1-96**}] [**-o numberofpasswordprompts***n*] [**-p** *port-num*] {*ip-addr* | *hostname*} [*command* **Example:** `Router# ssh`	Starts an encrypted session with a remote networking device.

Configuring an SNMP Notification Server over IPv6

Use an SNMP community string to define the relationship between the SNMP manager and the agent. The community string acts like a password to regulate access to the agent on the router. Optionally, you can specify one or more of the following characteristics associated with the string:

- An access list of IP addresses of the SNMP managers that are permitted to use the community string to gain access to the agent.
- A MIB view, which defines the subset of all MIB objects accessible to the given community.
- Read and write or read-only permission for the MIB objects accessible to the community.

You can configure one or more community strings. To remove a specific community string, use the **no snmp-server community** command.

The **snmp-server host** command specifies which hosts will receive SNMP notifications and whether you want the notifications sent as traps or inform requests. The **snmp-server enable traps** command globally

enables the production mechanism for the specified notification types (such as Border Gateway Protocol [BGP] traps, config traps, entity traps, and Hot Standby Router Protocol [HSRP] traps).

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **snmp-server community** *string* [**view** *view-name*] [**ro** | **rw**] [**ipv6** *nacl*] [*access-list-number*]
4. **snmp-server engineID remote** {*ipv4-ip-address*| *ipv6-address*}[**udp-port** *udp-port-number*] [**vrf** *vrf-name*] *engineid-string*
5. **snmp-server group** *group-name* {**v1** | **v2c** | **v3** {**auth** | **noauth** | **priv**}} [**context** *context-name*] [**read** *read-view*] [**write** *write-view*] [**notify** *notify-view*] [**access** [**ipv6** *named-access-list*]{*acl-number* | *acl-name*}]
6. **snmp-server host** {*hostname* | *ip-address*} [**vrf** *vrf-name*] [**traps** | **informs**] [**version** {**1** | **2c** | **3** [**auth** | **noauth** | **priv**]}] *community-string* [**udp-port** *port*] [*notification-type*]
7. **snmp-server user** *username group-name* [**remote** *host* [**udp-port** *port*]] {**v1** | **v2c** | **v3** [**encrypted**] [**auth** {**md5** | **sha**} *auth-password*]} [**access** [**ipv6** *nacl*] [**priv** {**des** | **3des** | **aes**{**128 192**| **256**}} *privpassword*] {*acl-number* | *acl-name*}]
8. **snmp-server enable traps** [*notification-type*] [**vrrp**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**snmp-server community** *string* [**view** *view-name*] [**ro**	**rw**] [**ipv6** *nacl*] [*access-list-number*] **Example:** `Router(config)# snmp-server community mgr view restricted rw ipv6 mgr2`	Defines the community access string.

Command or Action	Purpose
Step 4 **snmp-server engineID remote** {*ipv4-ip-address*| *ipv6-address*} [**udp-port** *udp-port-number*] [**vrf** *vrf-name*] *engineid-string* **Example:** `Router(config)# snmp-server engineID remote 3ffe:b00:c18:1::3/127 remotev6`	(Optional) Specifies the name of the remote SNMP engine (or copy of SNMP).
Step 5 **snmp-server group** *group-name* {**v1** | **v2c** | **v3** {**auth** | **noauth** | **priv**}} [**context** *context-name*] [**read** *read-view*] [**write** *write-view*] [**notify** *notify-view*] [**access** [**ipv6** *named-access-list*]{*acl-number* | *acl-name*}] **Example:** `Router(config)# snmp-server group public v2c access ipv6 public2`	(Optional) Configures a new SNMP group, or a table that maps SNMP users to SNMP views.
Step 6 **snmp-server host** {*hostname* | *ip-address*} [**vrf** *vrf-name*] [**traps** | **informs**] [**version** {**1** | **2c** | **3** [**auth** | **noauth** | **priv**]}] *community-string* [**udp-port** *port*] [*notification-type*] **Example:** `Router(config)# snmp-server host host1.com 2c vrf trap-vrf`	Specifies the recipient of an SNMP notification operation. Specifies whether you want the SNMP notifications sent as traps or informs, the version of SNMP to use, the security level of the notifications (for SNMPv3), and the recipient (host) of the notifications.
Step 7 **snmp-server user** *username group-name* [**remote** *host* [**udp-port** *port*]] {**v1** | **v2c** | **v3** [**encrypted**] [**auth** {**md5** | **sha**} *auth-password*]} [**access** [**ipv6** *nacl*] [**priv** {**des** | **3des** | **aes**{**128 192**| **256**}} *privpassword*] {*acl-number* | *acl-name*}] **Example:** `Router(config)# snmp-server user user1 bldg1 remote 3ffe:b00:c18:1::3/127 v2c access ipv6 public2`	(Optional) Configures a new user to an existing SNMP group. **Note** You cannot configure a remote user for an address without first configuring the engine ID for that remote host. This is a restriction imposed in the design of these commands; if you try to configure the user before the host, you will receive a warning message and the command will not be executed
Step 8 **snmp-server enable traps** [*notification-type*] [**vrrp** **Example:** `Router(config)# snmp-server enable traps bgp`	Enables sending of traps or informs, and specifies the type of notifications to be sent. • If a *notification-type* is not specified, all supported notification will be enabled on the router. • To discover which notifications are available on your router, enter the **snmp-server enable traps ?** command.

Configuring Cisco IOS XE IPv6 Embedded Management Components

Most IPv6 embedded management components are enabled automatically when IPv6 is enabled and do not need further configuration. To configure syslog over IPv6 or disable HTTP access to a router, refer to the tasks in the following sections:

Configuring Syslog over IPv6

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **logging host** {{*ip-address* | *hostname*} | {**ipv6** *ipv6-address* | *hostname*}} [**transport** {**udp** [**port** *port-number*] | **tcp** [**port** *port-number*] [**audit**]}] [**xml** | **filtered** [**stream** *stream-id*]] [**alarm** [*severity*]]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**logging host** {{*ip-address* | *hostname*} | {**ipv6** *ipv6-address* | *hostname*}} [**transport** {**udp** [**port** *port-number*] | **tcp** [**port** *port-number*] [**audit**]}] [**xml** | **filtered** [**stream** *stream-id*]] [**alarm** [*severity*]] **Example:** `Device(config)# logging host ipv6 AAAA:BBBB:CCCC:DDDD::FFFF`	Logs system messages and debug output to a remote host.

Disabling HTTP Access to an IPv6 Device

HTTP access over IPv6 is automatically enabled if an HTTP server is enabled and the device has an IPv6 address. If the HTTP server is not required, it should be disabled.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **no ip http server**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**no ip http server** **Example:** `Device(config)# no ip http server`	Disables HTTP access.

Configuration Examples for Implementing IPv6 for Network Management

Examples: Enabling Telnet Access to an IPv6 Device

The following examples provide information on how to enable Telnet and start a session to or from an IPv6 device. In the following example, the IPv6 address is specified as 2001:DB8:20:1::12, and the hostname is specified as cisco-sj. The **show host** command is used to verify this information.

```
Device# configure terminal
Device(config)# ipv6 host cisco-sj 2001:DB8:20:1::12
Device(config)# end
Device# show host
Default domain is not set
Name/address lookup uses static mappings
Codes:UN - unknown, EX - expired, OK - OK, ?? - revalidate
       temp - temporary, perm - permanent
```

```
       NA - Not Applicable None - Not defined
Host                 Port  Flags      Age Type   Address(es)
cisco-sj             None  (perm, OK)  0  IPv6 2001:DB8:20:1::12
```

To enable Telnet access to a device, create a vty interface and password:

```
Device(config)# line vty 0 4
password lab
login
```

To use Telnet to access the device, you must enter the password:

```
Device# telnet cisco-sj
Trying cisco-sj (2001:DB8:20:1::12)... Open
User Access Verification
Password:
cisco-sj
.
.
.
verification
```

It is not necessary to use the **telnet** command. Specifying either the hostname or the address is sufficient, as shown in the following examples:

```
Device# cisco-sj
```

or

```
Device# 2001:DB8:20:1::12
```

To display the IPv6 connected user (line 130) on the device to which you are connected, use the **show users** command:

```
Device# show users
    Line       User       Host(s)              Idle       Location
*  0 con 0                 idle                 00:00:00
 130 vty 0                 idle                 00:00:22   8800::3
```

Note that the address displayed is the IPv6 address of the source of the connection. If the hostname of the source is known (either through a domain name server [DNS] or locally in the host cache), then it is displayed instead:

```
Device# show users
    Line       User       Host(s)              Idle       Location
*  0 con 0                 idle                 00:00:00
 130 vty 0                 idle                 00:02:47   cisco-sj
```

If the user at the connecting device suspends the session with ^6x and then enters the **show sessions** command, the IPv6 connection is displayed:

```
Device# show sessions
Conn Host               Address             Byte  Idle Conn Name
*  1 cisco-sj 2001:DB8:20:1::12      0     0 cisco-sj
```

The Conn Name field shows the hostname of the destination only if it is known. If it is not known, the output might look similar to the following:

```
Device# show sessions
Conn Host               Address             Byte  Idle Conn Name
*  1 2001:DB8:20:1::12 2001:DB8:20:1::12      0     0 2001:DB8:20:1::12
```

Examples: Configuring an SNMP Notification Server over IPv6

The following example permits any SNMP to access all objects with read-only permission using the community string named public. The device also will send Border Gateway Protocol (BGP) traps to the IPv4 host 172.16.1.111 and IPv6 host 3ffe:b00:c18:1::3/127 using SNMPv1 and to the host 172.16.1.27 using SNMPv2c. The community string named public will be sent with the traps.

```
Device(config)# snmp-server community public
Device(config)# snmp-server enable traps bgp
Device(config)# snmp-server host 172.16.1.27 version 2c public
Device(config)# snmp-server host 172.16.1.111 version 1 public
Device(config)# snmp-server host 3ffe:b00:c18:1::3/127 public
```

Example: Associate an SNMP Server Group with Specified Views

In the following example, the SNMP context A is associated with the views in SNMPv2c group GROUP1 and the IPv6 named access list public2:

```
Device(config)# snmp-server context A
Device(config)# snmp mib community-map commA context A target-list commAVpn
Device(config)# snmp mib target list commAVpn vrf CustomerA
Device(config)# snmp-server view viewA ciscoPingMIB included
Device(config)# snmp-server view viewA ipForward included
Device(config)# snmp-server group GROUP1 v2c context A read viewA write viewA notify
access ipv6 public2
```

Example: Create an SNMP Notification Server

The following example configures the IPv6 host as the notification server:

```
Device> enable
Device# configure terminal
Device(config)# snmp-server community mgr view restricted rw ipv6 mgr2
Device(config)# snmp-server engineID remote 3ffe:b00:c18:1::3/127 remotev6
Device(config)# snmp-server group public v2c access ipv6 public2
Device(config)# snmp-server host host1.com 2c vrf trap-vrf
Device(config)# snmp-server user user1 bldg1 remote 3ffe:b00:c18:1::3/127 v2c access ipv6
public2
Device(config)# snmp-server enable traps bgp
Device(config)# exit
```

Additional References

Related Documents

Related Topic	Document Title
IPv6 supported features	" Start Here: Cisco IOS XE Software Release Specifics for IPv6 Features ," *Cisco IOS XE IPv6 Configuration Guide*
Basic IPv6 configuration tasks	" Implementing IPv6 Addressing and Basic Connectivity ," *Cisco IOS XE IPv6 Configuration Guide*

Related Topic	Document Title
IPv6 commands: complete command syntax, command mode, defaults, usage guidelines, and examples	*Cisco IOS IPv6 Command Reference*
SSH configuration information	*Cisco IOS Security Command Reference*
Cisco IOS master command list, all releases	Cisco IOS Master Command List, All Releases
IP SLAs for IPv6	• IP SLAs--Analyzing IP Service Levels Using the ICMP Echo Operation • IP SLAs--Analyzing IP Service Levels Using the TCP Connect Operation • IP SLAs--Analyzing IP Service Levels Using the UDP Echo Operation • IP SLAs--Analyzing IP Service Levels Using the UDP Jitter Operation • IP SLAs--Analyzing VoIP Service Levels Using the UDP Jitter Operation

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
None	To locate and download MIBs for selected platforms, Cisco software releases, and feature sets, use Cisco MIB Locator found at the following URL: http://www.cisco.com/go/mibs

RFCs

RFCs	Title
RFC 1350	*The TFTP Protocol (Revision 2)*
RFC 2732	*Format for Literal IPv6 Addresses in URLs*
RFC 3414	*User-based Security Model (USM) for version 3 of the Simple Network Management Protocol (SNMPv3)*

RFCs	Title
RFC 3484	*Default Address Selection for Internet Protocol version 6 (IPv6)*
RFC 4292	IP Forwarding Table MIB
RFC 4293	*Management Information Base for the Internet Protocol (IP)*

Technical Assistance

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

Feature Information for Implementing IPv6 for Network Management

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Table 14 Feature Information for Managing Cisco IOS XE Applications over IPv6

Feature Name	Releases	Feature Information
CNS Agents for IPv6	Cisco IOS XE Release 2.1	CNS configuration and event agents use a CNS configuration engine to provide methods for automating initial Cisco IOS device configurations, incremental configurations, and synchronized configuration updates, and the configuration engine reports the status of the configuration load as an event to which a network monitoring or workflow application can subscribe.
IP SLAs for IPv6	Cisco IOS XE Release 2.1	IP SLAs are supported for IPv6.
IPv6 for Config Logger	Cisco IOS XE Release 2.1	Config logger tracks and reports configuration changes.
IPv6--Syslog over IPv6	Cisco IOS XE Release 2.1	The Cisco IOS syslog process in IPv6 allows users to log syslog messages to external syslog servers and hosts with IPv6 addresses. The following command was modified by this feature: logging host
IPv6 Services--IP-FORWARD-MIB Support	Cisco IOS XE Release 2.1	A MIB is a database of objects that can be managed on a device. The managed objects, or variables, can be set or read to provide information on the network devices and interfaces.
IPv6 Services--IP-MIB Support	Cisco IOS XE Release 2.1	A MIB is a database of objects that can be managed on a device. The managed objects, or variables, can be set or read to provide information on the network devices and interfaces.
IPv6 Services--RFC 4293 IP-MIB (IPv6 only) and RFC 4292 IP-FORWARD-MIB (IPv6 only)	Cisco IOS XE Release 2.1	IP-FORWARD-MIB and IP-MIB were updated to RFC 4292 and RFC 4293 standards, respectively.
IPv6 Support for TCL	Cisco IOS XE Release 2.1	IPv6 supports TCL.

Feature Name	Releases	Feature Information
SNMP over IPv6	Cisco IOS XE Release 2.1	SNMP can be configured over IPv6 transport so that an IPv6 host can perform SNMP queries and receive SNMP notifications from a device running Cisco IOS IPv6. The following commands were modified by this feature: snmp-server community, snmp-server engineID remote, snmp-server group, snmp-server host, snmp-server user
SSH over an IPv6 Transport	Cisco IOS XE Release 2.1	SSH in IPv6 functions the same and offers the same benefits as SSH in IPv4--the SSH Server feature enables an SSH client to make a secure, encrypted connection to a Cisco router and the SSH Client feature enables a Cisco router to make a secure, encrypted connection to another Cisco router or to any other device running an SSH server. The following command was modified by this feature: **ssh**
Telnet Access over IPv6	Cisco IOS XE Release 2.1	The Telnet client and server in the Cisco IOS software support IPv6 connections. A user can establish a Telnet session directly to the router using an IPv6 Telnet client, or an IPv6 Telnet connection can be initiated from the router. The following commands were modified by this feature: **ipv6 access-class**, ipv6 host, **show host**, **show sessions**, **show users**, telnet
TFTP File Downloading for IPv6	Cisco IOS XE Release 2.1	IPv6 supports TFTP file downloading and uploading.
TFTP IPv6 Support	Cisco IOS XE Release 3.4S	TFTP uses UDP over IPv4 or IPv6 as its transport, and can work over IPv4 and IPv6 network layers.

Implementing IPv6 over MPLS

Multiprotocol Label Switching (MPLS) is deployed by many service providers in their IPv4 networks. Service providers want to introduce IPv6 services to their customers, but changes to their existing IPv4 infrastructure can be expensive and the cost benefit for a small amount of IPv6 traffic does not make economic sense. Several integration scenarios have been developed to leverage an existing IPv4 MPLS infrastructure and add IPv6 services without requiring any changes to the network backbone. This document describes how to implement IPv6 over MPLS.

Finding Feature Information

Your software release may not support all the features documented in this module. For the latest caveats and feature information, see Bug Search Tool and the release notes for your platform and software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the feature information table at the end of this module.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Prerequisites for Implementing IPv6 over MPLS

- This module assumes that you are familiar with IPv4. Refer to the publications referenced in the Prerequisites for Implementing IPv6 over MPLS, page 257 section for IPv4 configuration and command reference information.
- Before the IPv6 Provider Edge Router over MPLS (6PE) feature can be implemented, MPLS must be running over the core IPv4 network. If Cisco routers are used, Cisco Express Forwarding or distributed Cisco Express Forwarding must be enabled for both IPv4 and IPv6 protocols. This module assumes that you are familiar with MPLS.

Information About Implementing IPv6 over MPLS

Benefits of Deploying IPv6 over MPLS Backbones

IPv6 over MPLS backbones enables isolated IPv6 domains to communicate with each other over an MPLS IPv4 core network. This implementation requires only a few backbone infrastructure upgrades and no reconfiguration of core routers because forwarding is based on labels rather than the IP header itself, providing a very cost-effective strategy for the deployment of IPv6.

Additionally, the inherent Virtual Private Network (VPN) and MPLS traffic engineering (MPLS-TE) services available within an MPLS environment allow IPv6 networks to be combined into IPv4 VPNs or extranets over an infrastructure supporting IPv4 VPNs and MPLS-TE.

Limitations on using tunnels involve the manual configuring of a mesh of tunnels on the CE routers, creating scaling issues for large networks.

IPv6 on the Provider Edge Routers

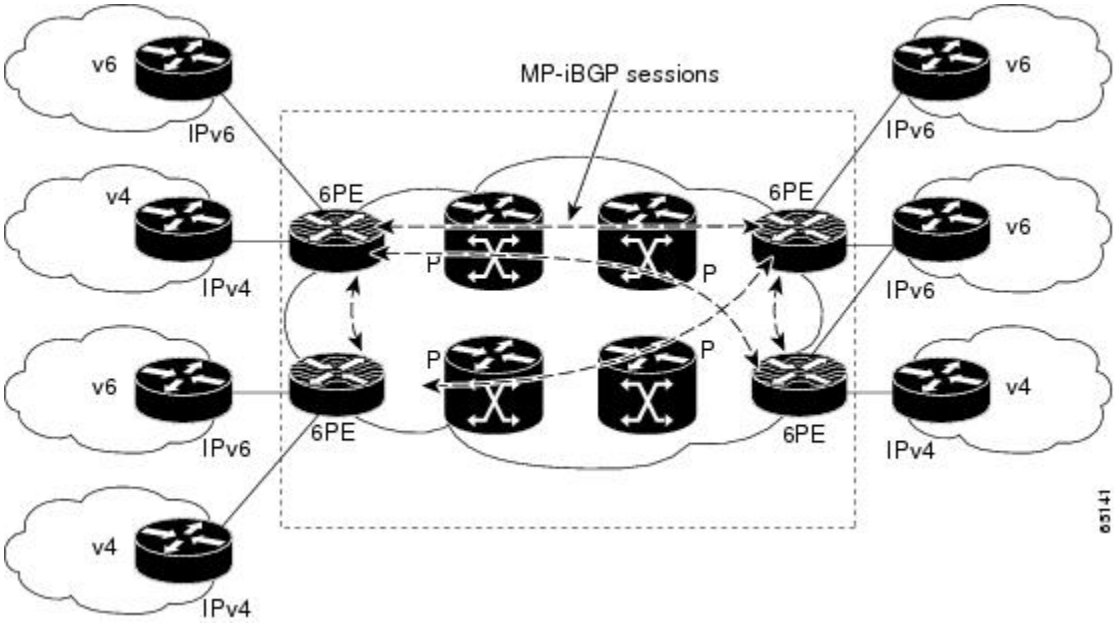
The Cisco implementation of IPv6 provider edge router over MPLS is called 6PE, and it enables IPv6 sites to communicate with each other over an MPLS IPv4 core network using MPLS label switched paths (LSPs). This feature relies on multiprotocol Border Gateway Protocol (BGP) extensions in the IPv4 network configuration on the provider edge (PE) router to exchange IPv6 reachability information in addition to an MPLS label for each IPv6 address prefix to be advertised. Edge routers are configured to be dual stack running both IPv4 and IPv6, and use the IPv4 mapped IPv6 address for IPv6 prefix reachability exchange.

A hierarchy of labels is imposed on the 6PE ingress router to keep the IPv6 traffic transparent to all the core routers. The top label provides connectivity inside the IPv4 MPLS core network and the label is distributed by Label Distribution Protocol (LDP), Tag Distribution Protocol (TDP), or Resource Reservation Protocol (RSVP). TDP and LDP can both be used for label distribution, but RSVP is used only in the context of MPLS-TE label exchange. The bottom label, automatically assigned to the IPv6 prefix of the destination, is distributed by multiprotocol BGP and used at each 6PE egress router for IPv6 forwarding.

In the figure below the 6PE routers are configured as dual stack routers able to route both IPv4 and IPv6 traffic. Each 6PE router is configured to run LDP, TDP, or RSVP (if traffic engineering is configured) to bind the IPv4 labels. The 6PE routers use multiprotocol BGP to exchange reachability information with the other 6PE devices within the MPLS domain, and to distribute aggregate IPv6 labels between them. All 6PE and core routers--P routers in Figure 3--within the MPLS domain share a common IPv4 Interior Gateway

Protocol (IGP) such as Open Shortest Path First (OSPF) or Integrated Intermediate System-to-Intermediate System (IS-IS).

Figure 25 6PE Router Topology



The interfaces on the 6PE routers connecting to the CE router can be configured to forward IPv6 traffic, IPv4 traffic, or both types of traffic depending on the customer requirements. 6PE routers advertise IPv6 reachability information learned from their 6PE peers over the MPLS cloud. Service providers can delegate an IPv6 prefix from their registered IPv6 prefixes over the 6PE infrastructure; otherwise, there is no impact on the CE router.

The P routers in the core of the network are not aware that they are switching IPv6 packets. Core routers are configured to support MPLS and the same IPv4 IGP as the PE routers to establish internal reachability inside the MPLS cloud. Core routers also use LDP, TDP, or RSVP for binding IPv4 labels. Implementing the Cisco 6PE feature does not have any impact on the MPLS core devices.

Within the MPLS network, IPv6 traffic is forwarded using label switching, making the IPv6 traffic transparent to the core of the MPLS network. No IPv6 over IPv4 tunnels or Layer 2 encapsulation methods are required.

- 6PE Multipath, page 259

6PE Multipath

Internal and external BGP multipath for IPv6 allows the IPv6 device to load balance between several paths (for example, the same neighboring autonomous system or subautonomous system, or the same metric) to reach its destination. The 6PE multipath feature uses MP-iBGP to distribute IPv6 routes over the MPLS IPv4 core network and to attach an MPLS label to each route.

When MP-iBGP multipath is enabled on the 6PE device, all labeled paths are installed in the forwarding table with MPLS information (label stack) when MPLS information is available. This functionality enables 6PE to perform load balancing.

How to Implement IPv6 over MPLS

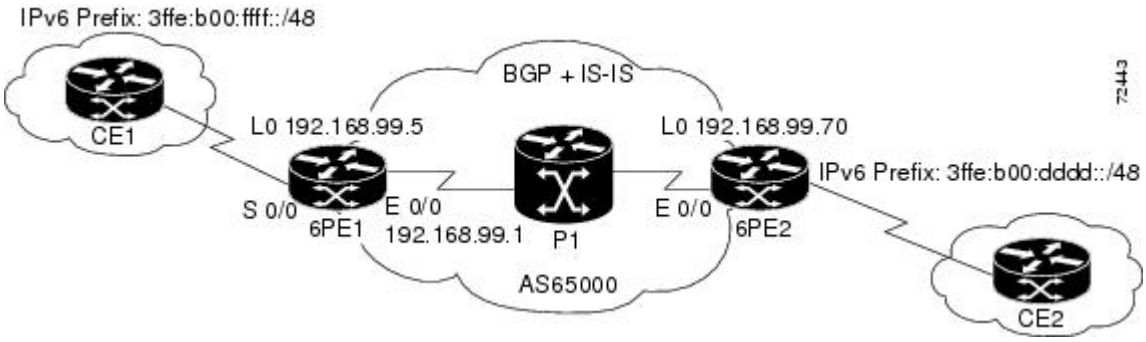
Deploying IPv6 on the Provider Edge Routers (6PE)

Specifying the Source Address Interface on a 6PE Router

Two configuration tasks using the network shown in the figure below are required at the 6PE1 router to enable the 6PE feature.

The customer edge router--CE1 in the figure below--is configured to forward its IPv6 traffic to the 6PE1 router. The P1 router in the core of the network is assumed to be running MPLS, a label distribution protocol, an IPv4 IGP, and Cisco Express Forwarding or distributed Cisco Express Forwarding, and does not require any new configuration to enable the 6PE feature.

Figure 26 6PE Configuration Example



- The 6PE routers--the 6PE1 and 6PE2 routers in the figure below--must be members of the core IPv4 network. The 6PE router interfaces attached to the core network must be running MPLS, the same label distribution protocol, and the same IPv4 IGP, as in the core network.
- The 6PE routers must also be configured to be dual stack to run both IPv4 and IPv6.

Note The following restrictions apply when implementing the IPv6 Provider Edge Router over MPLS (6PE) feature:

- Core MPLS routers are supporting MPLS and IPv4 only, so they cannot forward or create any IPv6 Internet Control Message Protocol (ICMP) messages.
- Load balancing ability is not provided by Cisco 6PE between an MPLS path and an IPv6 path. If both are available, the MPLS path is always preferred. Load balancing between two MPLS paths is possible.
- BGP multipath is not supported for Cisco 6PE routes. If two BGP peers advertise the same prefix with an equal cost, Cisco 6PE will use the last route to cross the MPLS core.
- 6PE feature is not supported over tunnels other than RSVP-TE tunnels.

Perform this task to specify the interface from which locally generated packets take their source IPv6 address.

>

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ipv6 unicast-routing**
4. **ipv6 cef**
5. **interface** *type number*
6. **ipv6 address** *ipv6-address /prefix-length | prefix-name sub-bits/prefix-length*

DETAILED STEPS

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

	Command or Action	Purpose
Step 4	**ipv6 cef** **Example:** `Router(config)# ipv6 cef`	Enables IPv6 Cisco Express Forwarding.
Step 5	**interface** *type number* **Example:** `Router(config)# interface GigabitEthernet 0/0/0`	Specifies an interface type and number and enters interface configuration mode. • In the context of this feature, the interface to be configured is the interface communicating with the CE router.
Step 6	**ipv6 address** *ipv6-address /prefix-length | prefix-name sub-bits/prefix-length* **Example:** `Router(config-if)# ipv6 address 2001:DB8:FFFF::2/64`	Configures an IPv6 address based on an IPv6 general prefix and enable IPv6 processing on an interface.

Binding and Advertising the 6PE Label to Advertise Prefixes

Perform this task to enable the binding and advertising of aggregate labels when advertising IPv6 prefixes to a specified BGP neighbor.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **router bgp** *as-number*
4. **no bgp default ipv4-unicast**
5. **neighbor** {*ip-address* | *ipv6-address* | *peer-group-name*} **remote-as** *as-number*
6. **neighbor** { *ip-address* | *ipv6-address* | *peer-group-name*} **update-source** *interface-type interface-number*
7. **address-family ipv6** [**unicast**]
8. **neighbor** {*ip-address* | *peer-group-name*| *ipv6-address*} **activate**
9. **neighbor** {*ip-address* | *ipv6-address*} **send-label**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**router bgp** *as-number* **Example:** `Router(config)# router bgp 65000`	Enters router configuration mode for the specified routing process.
Step 4	**no bgp default ipv4-unicast** **Example:** `Router(config-router)# no bgp default ipv4-unicast`	Disables the IPv4 unicast address family for the BGP routing process specified in the previous step. **Note** Routing information for the IPv4 unicast address family is advertised by default for each BGP routing session configured with the **neighbor remote-as** command unless you configure the **no bgp default ipv4-unicast**command before configuring the **neighbor remote-as** command.
Step 5	**neighbor** {*ip-address* | *ipv6-address* | *peer-group-name*} **remote-as** *as-number* **Example:** `Router(config-router)# neighbor 192.168.99.70 remote-as 65000`	Adds the IP address of the neighbor in the specified autonomous system to the BGP neighbor table of the local router.
Step 6	**neighbor** { *ip-address* | *ipv6-address* | *peer-group-name*} **update-source** *interface-type interface-number* **Example:** `Router(config-router)# neighbor 192.168.99.70 update-source Loopback 0`	Specifies the interface whose IPv4 address is to be used as the source address for the peering. • In the context of this task, the interface must have an IPv4 address with a 32-bit mask configured. Use of a loopback interface is recommended. This address is used to determine the IPv6 next hop by the peer 6PE.

	Command or Action	Purpose
Step 7	**address-family ipv6** [**unicast**] **Example:** `Router(config-router)# address-family ipv6`	Specifies the IPv6 address family and enters address family configuration mode. • The **unicast** keyword specifies the IPv6 unicast address family. By default, the router is placed in configuration mode for the IPv6 unicast address family if the **unicast** keyword is not specified with the **address-family ipv6** command.
Step 8	**neighbor** {*ip-address* | *peer-group-name*| *ipv6-address*} **activate** **Example:** `Router(config-router-af)# neighbor 192.168.99.70 activate`	Enables the neighbor to exchange prefixes for the IPv6 address family with the local router.
Step 9	**neighbor** {*ip-address* | *ipv6-address*} **send-label** **Example:** `Router(config-router-af)# neighbor 192.168.99.70 send-label`	Advertises the capability of the router to send MPLS labels with BGP routes. • In IPv6 address family configuration mode this command enables binding and advertisement of aggregate labels when advertising IPv6 prefixes in BGP.

Configuring iBGP Multipath Load Sharing

Perform this task to configure iBGP multipath load sharing and control the maximum number of parallel iBGP routes that can be installed in a routing table.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **router bgp** *as-number*
4. **maximum-paths ibgp** *number-of-paths*

DETAILED STEPS

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

	Command or Action	Purpose
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**router bgp** *as-number* **Example:** `Router(config)# router bgp 65000`	Enters router configuration mode for the specified routing process.
Step 4	**maximum-paths ibgp** *number-of-paths* **Example:** `Router(config-router)# maximum-paths ibgp 3`	Controls the maximum number of parallel iBGP routes that can be installed in a routing table.

Verifying 6PE Configuration and Operation

SUMMARY STEPS

1. **show bgp ipv6** {**unicast** | **multicast**} [*ipv6-prefix* / *prefix-length*] [**longer-prefixes**] [**labels**
2. **show bgp ipv6** {**unicast** | **multicast**} **neighbors** [*ipv6-address*] [**received-routes** | **routes** | **flap-statistics** | **advertised-routes** | **paths** *regular-expression* | **dampened-routes**
3. **show mpls forwarding-table** [*network*{*mask*| *length*}| **labels** *label*[- *label*] | **interface** *interface*| **nexthop** *address*| **lsp-tunnel**[*tunnel-id*]] [**vrf** *vrf-name*] [**detail**]
4. **show ipv6 cef** [*ipv6-prefix* / *prefix-length*] | [*interface-type interface-number*] [**longer-prefixes** | **similar-prefixes** | **detail** | **internal** | **platform** | **epoch** | **source**]]
5. **show ipv6 route** [*ipv6-address* | *ipv6-prefix*/*prefix-length* | *protocol* | *interface-type interface-number*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**show bgp ipv6** {**unicast**	**multicast**} [*ipv6-prefix* / *prefix-length*] [**longer-prefixes**] [**labels** **Example:** `Router> show bgp ipv6 unicast 2001:DB8:DDDD::/48`	(Optional) Displays entries in the IPv6 BGP routing table. • In this example, information about the IPv6 route for the prefix 2001:DB8:DDDD::/48 is displayed.

Command or Action	Purpose
Step 2 **show bgp ipv6** {**unicast** | **multicast**} **neighbors** [*ipv6-address*] [**received-routes** | **routes** | **flap-statistics** | **advertised-routes** | **paths** *regular-expression* | **dampened-routes** **Example:** `Router> show bgp ipv6 neighbors unicast 192.168.99.70`	(Optional) Displays information about IPv6 BGP connections to neighbors. • In this example, information including the IPv6 label capability is displayed for the BGP peer at 192.168.99.70.
Step 3 **show mpls forwarding-table** [*network*{*mask*| *length*}| **labels** *label*[- *label*] | **interface** *interface*| **nexthop** *address*| **lsp-tunnel**[*tunnel-id*]] [**vrf** *vrf-name*] [**detail**] **Example:** `Router> show mpls forwarding-table`	(Optional) Displays the contents of the MPLS Forwarding Information Base (FIB). • In this example, information linking the MPLS label with IPv6 prefixes is displayed where the labels are shown as aggregate and the prefix is shown as IPv6.
Step 4 **show ipv6 cef** [*ipv6-prefix* / *prefix-length*] | [*interface-type interface-number*] [**longer-prefixes** | **similar-prefixes** | **detail** | **internal** | **platform** | **epoch** | **source**]] **Example:** `Router> show ipv6 cef 2001:DB8:DDDD::/64`	(Optional) Displays FIB entries based on IPv6 address information. • In this example, label information from the Cisco Express Forwarding table for prefix 2001:DB8:DDDD::/64 is displayed.
Step 5 **show ipv6 route** [*ipv6-address* | *ipv6-prefix*/*prefix-length* | *protocol* | *interface-type interface-number* **Example:** `Router> show ipv6 route`	(Optional) Displays the current contents of the IPv6 routing table.

• Output Examples, page 266

Output Examples

Sample Output from the show bgp ipv6 Command

In the following example, output information about an IPv6 route is displayed using the **show bgp ipv6**command with an IPv6 prefix:

```
Router# show bgp ipv6 2001:DB8:DDDD::/48
BGP routing table entry for 2001:DB8:DDDD::/48, version 15
Paths: (1 available, best #1, table Global-IPv6-Table)
  Not advertised to any peer
  Local
```

```
    ::FFFF:192.168.99.70 (metric 20) from 192.168.99.70 (192.168.99.70)
      Origin IGP, localpref 100, valid, internal, best
```

Sample Output from the show bgp ipv6 neighbors Command

In the following example, output information about a BGP peer including the "IPv6 label" capability is displayed using the **show bgp ipv6 neighbors**command with an IP address:

```
Router# show bgp ipv6 neighbors 192.168.99.70
BGP neighbor is 192.168.99.70, remote AS 65000, internal link
  BGP version 4, remote router ID 192.168.99.70
  BGP state = Established, up for 00:05:17
  Last read 00:00:09, hold time is 0, keepalive interval is 60 seconds
  Neighbor capabilities:
    Route refresh: advertised and received
    Address family IPv6 Unicast: advertised and received
    ipv6 MPLS Label capability: advertised and received
  Received 54 messages, 0 notifications, 0 in queue
  Sent 55 messages, 1 notifications, 0 in queue
  Default minimum time between advertisement runs is 5 seconds
 For address family: IPv6 Unicast
  BGP table version 21880, neighbor version 21880
  Index 1, Offset 0, Mask 0x2
  Route refresh request: received 0, sent 0
  77 accepted prefixes consume 4928 bytes
  Prefix advertised 4303, suppressed 0, withdrawn 1328
  Number of NLRIs in the update sent: max 1, min 0
```

Sample Output from the show mpls forwarding-table Command

In the following example, output information linking the MPLS label with prefixes is displayed using the **show mpls forwarding-table**command. If the 6PE feature is configured, the labels are aggregated because there are several prefixes for one local label, and the prefix column contains "IPv6" instead of a target prefix.

```
Router# show mpls forwarding-table
Local Outgoing      Prefix            Bytes tag Outgoing       Next Hop
tag   tag or VC     or Tunnel Id      switched  interface
16    Aggregate     IPv6              0
17    Aggregate     IPv6              0
18    Aggregate     IPv6              0
19    Pop tag       192.168.99.64/30  0         GE0/0         point2point
20    Pop tag       192.168.99.70/32  0         GE0/0         point2point
21    Pop tag       192.168.99.200/32 0         GE0/0         point2point
22    Aggregate     IPv6              5424
23    Aggregate     IPv6              3576
24    Aggregate     IPv6              2600
```

Sample Output from the show bgp ipv6 Command

In the following example, output information about the top of the stack label with label switching information is displayed using the **show bgp ipv6**command with the **labels** keyword:

```
Router# show bgp ipv6 labels
Network              Next Hop              In tag/Out tag
2001:DB8:DDDD::/64  ::FFFF:192.168.99.70   notag/20
```

Sample Output from the show ipv6 cef Command

In the following example, output information about labels from the Cisco Express Forwarding table is displayed using the **show ipv6 cef**command with an IPv6 prefix:

```
Router# show ipv6 cef 2001:DB8:DDDD::/64
2001:DB8:DDDD::/64
```

```
      nexthop ::FFFF:192.168.99.70
      fast tag rewrite with Se0/0, point2point, tags imposed {19 20}
```

Sample Output from the show ipv6 route Command

In the following example, output information from the IPv6 routing table is displayed using the **show ipv6 route**command. The output shows the IPv6 MPLS virtual interface as the output interface of IPv6 routes forwarded across the MPLS cloud. This example shows output from the 6PE1 router.

The 6PE2 router has advertised the IPv6 prefix of 2001:DB8:dddd::/48 configured for the CE2 router and the next-hop address is the IPv4-compatible IPv6 address ::ffff:192.168.99.70, where 192.168.99.70 is the IPv4 address of the 6PE2 router.

```
Router# show ipv6 route
IPv6 Routing Table - 10 entries
Codes: C - Connected, L - Local, S - Static, R - RIP, B - BGP
I1 - ISIS L1, I2 - ISIS L2, IA - ISIS interarea
B 2001:DB8:DDDD::/64 [200/0]
  via ::FFFF:192.168.99.70, IPv6-mpls
B 2001:DB8:DDDD::/64 [200/0]
  via ::FFFF:192.168.99.70, IPv6-mpls
L 2001:DB8:FFFF::1/128 [0/0]
  via ::, GigabitEthernet0/0/0
C 2001:DB8:FFFF::/64 [0/0]
  via ::, GigabitEthernet0/0/0
S 2001:DB8:FFFF::/48 [1/0]
  via 2001:DB8:B00:FFFF::2, GigabitEthernet0/0/0
```

Configuration Examples for IPv6 over MPLS

The following examples show 6PE configuration examples.

Example: Provider Edge Router

The 6PE router is configured for both IPv4 and IPv6 traffic. GigabitEthernet interface 0/0/0 is configured with an IPv4 address and is connected to a router in the core of the network. Integrated IS-IS and TDP configurations on this router are similar to the P1 router.

Router 6PE1 exchanges IPv6 routing information with another 6PE router using internal BGP (iBGP) established over an IPv4 connection so that all the **neighbor** commands use the IPv4 address of the 6PE2 router. All the BGP peers are within autonomous system 65000, so synchronization with IGP is turned off for IPv4. In IPv6 address family configuration mode, synchronization is disabled by default.

IPv6 and Cisco Express Forwarding for IPv6 are enabled, the 6PE2 neighbor is activated, and aggregate label binding and advertisement is enabled for IPv6 prefixes using the **neighbor send-label** command. Connected and static IPV6 routes are redistributed using BGP. If IPv6 packets are generated in the local router, the IPv6 address for MPLS processing will be the address of loopback interface 0.

In the following example, serial interface 0/0 connects to the customer and the IPv6 prefix delegated to the customer is 2001:DB8:ffff::/48, which is determined from the service provider IPv6 prefix. A static route is configured to route IPv6 packets between the 6PE route and the CE router.

```
ip cef
ipv6 cef
ipv6 unicast-routing
!
```

```
mpls ipv6 source-interface Loopback0
tag-switching tdp router-id Loopback0
!
interface Loopback0
 ip address 192.168.99.5 255.255.255.255
 ipv6 address 2001:DB8:1000:1::1/64
!
interface GigabitEthernet0/0/0
 description to_P_router
 ip address 192.168.99.1 255.255.255.252
 ip router isis
 tag-switching ip
!
interface GigabitEthernet0/1/0
 description to_CE_router
 no ip address
 ipv6 address 2001:DB8:FFFF::1/64
!
router isis
 passive-interface Loopback0
 net 49.0001.1921.6809.9005.00
!
router bgp 65000
 no bgp default ipv4-unicast
 bgp log-neighbor-changes
 neighbor 192.168.99.70 remote-as 65000
 neighbor 192.168.99.70 description to_6PE2
 neighbor 192.168.99.70 update-source Loopback0
 !
 address-family ipv6
 neighbor 192.168.99.70 activate
 neighbor 192.168.99.70 send-label
 network 2001:DB8:FFFF::/48
 exit-address-family
!
ipv6 route 2001:DB8:FFFF::/48 GigabitEthernet0/0/0 2001:DB8:FFFF::2
```

Example: Core Router

In the following example, the router in the core of the network is running MPLS, IS-IS, and IPv4 only. The GigabitEthernet interfaces are configured with IPv4 address and are connected to the 6PE routers. IS-IS is the IGP for this network and the P1 and 6PE routers are in the same IS-IS area 49.0001. TDP and tag switching are enabled on both the GigabitEthernet interfaces. Cisco Express Forwarding is enabled in global configuration mode.

```
ip cef
!
tag-switching tdp router-id Loopback0
!
interface Loopback0
 ip address 192.168.99.200 255.255.255.255
!
interface GigabitEthernet0/0/0
 description to_6PE1
 ip address 192.168.99.2 255.255.255.252
 ip router isis
 tag-switching ip
!
interface GigabitEthernet0/1/0
 description to_6PE2
 ip address 192.168.99.66 255.255.255.252
 ip router isis
 tag-switching ip
router isis
 passive-interface Loopback0
 net 49.0001.1921.6809.9200.00
```

Additional References

Related Documents

Related Topic	Document Title
IPv6 supported feature list	" Start Here: Cisco IOS XE Software Release Specifics for IPv6 Features ," *Cisco IOS XE IPv6 Configuration Guide*
IPv6 commands: complete command syntax, command mode, defaults, usage guidelines, and examples	*Cisco IOS IPv6 Command Reference*
Cisco IOS master command list, all releases	Cisco IOS Master Command List, All Releases

MIBs

MIBs	MIBs Link
None	To locate and download MIBs for selected platforms, Cisco software releases, and feature sets, use Cisco MIB Locator found at the following URL: http://www.cisco.com/go/mibs

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

Feature Information for Implementing IPv6 over MPLS

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Table 15 Feature Information for Implementing IPv6 over MPLS

Feature Name	Releases	Feature Information
IPv6 Switching--Provider Edge Router over MPLS (6PE)	Cisco IOS XE Release 3.1S	The Cisco implementation of IPv6 provider edge router over MPLS enables IPv6 sites to communicate with each other over an MPLS IPv4 core network using MPLS LSPs. The following commands were modified for this release: **ipv6 cef**, **neighbor activate**, **neighbor remote-as**, **neighbor send-label**, **neighbor update-source**, **router bgp**, **show bgp ipv6**, **show bgp ipv6 neighbors**, **show ipv6 cef**, **show ipv6 route**, **show mpls forwarding-table**.
6PE Multipath	Cisco IOS XE Release 3.1S	The 6PE multipath feature uses multiprotocol internal BGP (MP-iBGP) to distribute IPv6 routes over the MPLS IPv4 core network and to attach an MPLS label to each route. The following commands were modified for this release: **maximum-paths ibgp**.

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.

Implementing IPv6 VPN over MPLS

The Border Gateway Protocol over Multiprotocol Label Switching VPN feature is an implementation of the provider edge (PE)-based VPN model. In principle, there is no difference between IPv4 and IPv6 VPNs. In both IPv4 and IPv6, multiprotocol Border Gateway Protocol (BGP) is the center of the Multiprotocol Label Switching (MPLS) VPN for IPv6 (VPNv6) architecture. It is used to distribute IPv6 routes over the service provider backbone, using the same procedures to work with overlapping addresses, redistribution policies, and scalability issues.

Finding Feature Information

Your software release may not support all the features documented in this module. For the latest caveats and feature information, see Bug Search Tool and the release notes for your platform and software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the feature information table at the end of this module.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Prerequisites for Implementing IPv6 VPN over MPLS

Your network must be running the following Cisco IOS XE services before you configure IPv6 VPN operation:

- MPLS in provider backbone routers
- MPLS with VPN code in provider routers with VPN PE routers
- BGP in all routers providing a VPN service
- Cisco Express Forwarding switching in every MPLS-enabled router
- Class of Service (CoS) feature

Restrictions for Implementing IPv6 VPN over MPLS

6VPE supports an MPLS IPv4-signaled core. An MPLS IPv6-signaled core is not supported.

Information About Implementing IPv6 VPN over MPLS

IPv6 VPN over MPLS Overview

Multiprotocol BGP is the center of the MPLS IPv6 VPN architecture in both IPv4 and IPv6. It is used to distribute IPv6 routes over the service provider backbone, using the same procedures to work with overlapping addresses, redistribution policies, and scalability issues.

Although IPv6 should not have overlapping address space, IPv6 addresses are prepended with a route distinguisher (RD). A network layer reachability information (NLRI) 3-tuple format (which contains length, IPv6 prefix, and label) is defined to distribute these routes using multiprotocol BGP. The extended community attribute (for example., the route target) is used to control redistribution of routing information by tagging exported routes and filtering imported ones.

For scalability, route reflectors can be used to concentrate routing paths and avoid a full PE mesh. BGP features in IPv6, such as route refresh, automatic route filtering, and outbound route filtering, help reduce the number of routes held in each PE. This document focuses on the following differences between IPv6 and IPv4:

- Creation of a new multiprotocol BGP IPv6 VPN address family and specification of a IPv6 VPN address format
- Specification of a new IPv6 VPN NLRI
- Specification of BGP next-hop encoding when the device has an IPv4-based MPLS core

Some IPv6 VPN features, such as interprovider and Carrier Supporting Carrier (CSC) topologies, are specific to BGP-MPLS IPv6 VPN. Others, such as the link between Autonomous System Boundary Routers (ASBRs), might support IPv4 only, IPv6 only, or both, regardless of the address family being transported.

Addressing Considerations for IPv6 VPN over MPLS

Regardless of the VPN model deployed, an addressing plan must be defined for the VPN that allows hosts to communicate with other sites using one site within one VPN, as well as with public resources.

VPN IPv4 sites often use private addressing for their addressing plan. These addresses do not need to be registered, and they are not routable on the public network. Whenever a host within a private site needs to access a public domain, it goes through a device that finds a public address on its behalf. With IPv4, this can be a network address translator or an application proxy.

Given the larger address space available with IPv6, the easiest approach to IPv6 addressing is to use IPv6 global addresses for the private addressing plan. Another approach is to use unique local addresses (ULAs).

ULAs are easy to filter at site boundaries based on their scope. ULAs are also Internet service provider (ISP)-independent and can be used for communications inside a site without any permanent or intermittent Internet connectivity.

In 6VPE, ULAs are treated as regular global addresses. The device configuration filters ULA prefixes to prevent them from appearing in the public domain. Link-local addresses on the peer will not be announced by BGP (IPv6 or IPv6 VPN) speakers.

A host within a private site that needs to access a public domain can do so through an IPv6 application proxy (such as a web proxy for accessing web pages), which accesses the public resource on the host's behalf with a global routable address, or the host can use a public address of its own. In the latter case, if ULAs have been deployed, the IPv6 host also is configured with a routable global address. A source address selection algorithm is used to select one or the other, based on the destination address.

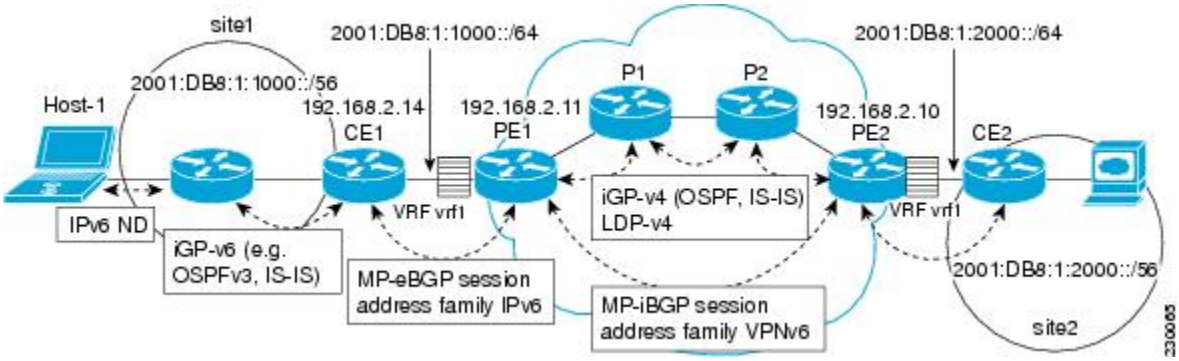
Basic IPv6 VPN over MPLS Functionality

IPv6 VPN takes advantage of the coexistence between IPv6 and IPv4 by leveraging an existent MPLS IPv4 core network:

IPv6 VPN Architecture Overview

The figure below illustrates the important aspects of the IPv6 VPN architecture.

Figure 27 Simple IPv6 VPN Architecture



The CE devices are connected to the provider's backbone using PE devices. The PE devices are connected using provider (P1 and P2 in the figure above) devices. The provider (P) devices are unaware of VPN routes, and, in the case of 6VPE, might support only IPv4. Only PE devices perform VPN-specific tasks. For 6VPE, the PE devices are dual-stack (IPv4 and IPv6) devices.

The routing component of the VPN operation is divided into core routing and edge routing. Core routing, which involves PE devices and P devices, typically is performed by an IPv4 Interior Gateway Protocol (IGP) such as Open Shortest Path First (OSPF) or Intermediate System-to-Intermediate System (IS-IS). In the figure above, the IGP distributes only routes internal to the provider's autonomous system. The core routing enables connectivity among P and PE devices.

Edge routing takes place in two directions: routing between PE pairs and routing between a PE and a CE. Routing between PE pairs is achieved using multiprotocol internal BGP (iBGP) using the IPv6 VPN address family. This method distributes routes learned from CEs through PE-CE routing, using appropriate route export policies at the ingress PE device and appropriate route import policies at the egress PE device.

Routing between the CE and its PE is achieved using a routing protocol that is VPN routing and forwarding (VRF) aware. Static routes, external BGP (eBGP), and Enhanced Interior Gateway Routing Protocol (EIGRP) are VRF-instance aware. In the figure above, eBGP is used between the CE (CE1) and the PE (PE1). At the same time, the CE runs an IPv6 IGP within the VPN site (site1 in the figure above). The CE redistributes IGP routes into multiprotocol-eBGP address family IPv6. At the PE, these routes are installed in the VRF named vrf1, and forwarded to the remote PEs (PE2 in the figure above), according to export policies defined for this VRF.

IPv6 VPN Next Hop

When the device announces a prefix using the MP_REACH_NLRI attribute, the MP-BGP running on one PE inserts a BGP next hop in the update message sent to a remote PE. This next hop is either propagated from the received update (for instance, if the PE is a route reflector), or it is the address of the PE sending the update message (the egress PE).

For the IPv6 VPN address family, the next hop must be an IPv6 VPN address, regardless of the nature of the network between the PE speakers. Because the RD has no significance (the address is not part of any VPN), it is set to 0. If the provider network is a native IPv6 network, the remaining part of the next hop is the IPv6 address of the egress PE. Otherwise, it is an IPv4 address used as an IPv6-mapped address (for example, ::FFFF:IPv4-address).

MPLS Forwarding

When it receives IPv6 traffic from one customer site, the ingress PE device uses MPLS to tunnel IPv6 VPN packets over the backbone toward the egress PE device identified as the BGP next hop. The ingress PE device prepends the IPv6 packets with the outer and inner labels before putting the packet on the egress interface.

Under normal operation, a P device along the forwarding path does not look inside the frame beyond the first label. The P device either swaps the incoming label with an outgoing one or removes the incoming label if the next device is a PE device. Removing the incoming label is called penultimate hop popping. The remaining label (BGP label) is used to identify the egress PE interface toward the customer site. The label also hides the protocol version (IPv6) from the last P device, which it would otherwise need to forward an IPv6 packet.

A P device is ignorant of the IPv6 VPN routes. The IPv6 header remains hidden under one or more MPLS labels. When the P device receives an MPLS-encapsulated IPv6 packet that cannot be delivered, it has two options. If the P device is IPv6 aware, it exposes the IPv6 header, builds an Internet Control Message Protocol (ICMP) for IPv6 message, and sends the message, which is MPLS encapsulated, to the source of the original packet. If the P device is not IPv6 aware, it drops the packet.

- 6VPE over GRE Tunnels, page 276

6VPE over GRE Tunnels

In some Cisco software releases, the ingress PE device uses IPv4 generic routing encapsulation (GRE) tunnels combined with 6VPE over MPLS to tunnel IPv6 VPN packets over the backbone toward the egress PE device identified as the BGP next hop.

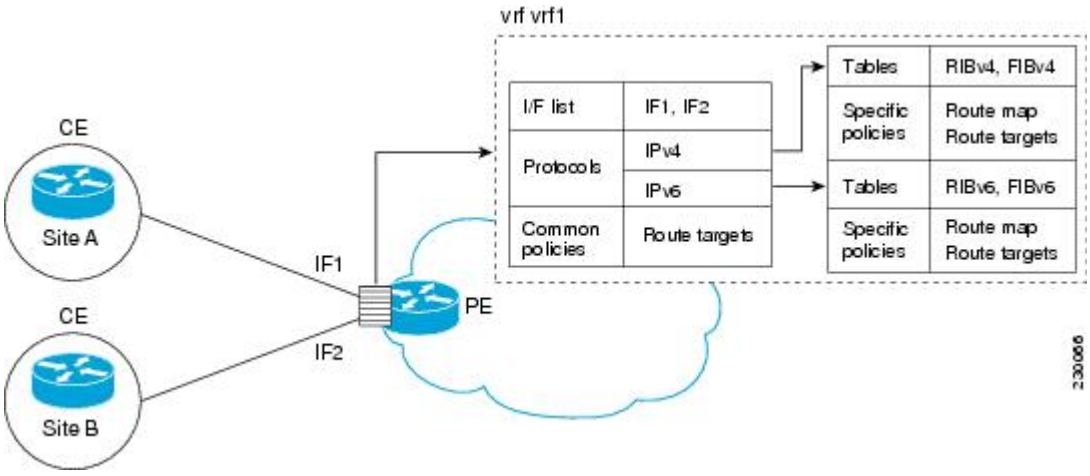
VRF Concepts

A virtual routing and forwarding (VRF) entity works with a private customer-specific Routing Information Base (RIB) and Forwarding Information Base (FIB). Although IPv4 and IPv6 routing tables are distinct, it is convenient for the two protocols to share the same VRF for a specific customer.

IPv6 VPN customers are likely to be existing VPNv4 customers that are either deploying dual-stack hosts and devices or shadowing some of their IPv4 infrastructure with IPv6 nodes. Several deployment models are possible. Some customers use separate logical interfaces for IPv4 and IPv6 and define separate VRFs on each. Although this approach provides flexibility to configure separate policies for IPv4 and IPv6, it prevents sharing the same policy. Another approach, the multiprotocol VRF, keeps a single VRF on the PE-CE interface, and enables it for IPv4, IPv6, or both. It is then possible to define common or separate policies for each IP version. With this approach, a VRF is better defined as the set of tables, interfaces, and policies found at the PE, and is used by sites of a particular VPN connected to this PE.

The figure below illustrates the multiprotocol VRF, in which the VRF named vrf1is enabled for both IPv4 and IPv6 and is associated with two interfaces (IF1, IF2), two sets of tables (IPv4 RIB and FIB and IPv6 RIB and FIB), and a set of common or distinct policies.

Figure 28 Multiprotocol VRF



IPv6 VPN Scalability

PE-based VPNs such as BGP-MPLS IPv6 VPN scale better than CE-based VPNs. A network designer must consider scaling when designing the network. The following points need to be considered:

- Routing table size, which includes the size of VRF tables and BGP tables
- Number of BGP sessions, which grows as a square number of PEs

Routing table size concerns occur with PEs that handle many customer sites. Not only do these PEs have one RIB and FIB per connected customer, but also the PEs' BGP tables, which total all entries from individual VRFs, grow accordingly. Another scalability problem occurs when the number of PEs in the provider network grows beyond a certain level. Assuming that a significant number of sites belonging to the same VPN are spread over many PEs, the number of multiprotocol BGP sessions may rapidly become prohibitive: (n -1) x n /2, where n is the number of PEs.

The following features are included in IPv6 VPN over MPLS:

- Route refresh and automatic route filtering--Limits the size of routing tables, because only routes imported into a VRF are kept locally. When the import policy changes, a route refresh can be sent to query a retransmission of routing updates.
- Outbound route filtering (ORF)--Allows the ingress PE to advertise filters to the egress PE so that updates are not sent unnecessarily over the network.
- Route reflectors--Route reflectors (RRs) are iBGP peers that propagate iBGP routes learned from other iBGP peers. RRs are used to concentrate iBGP sessions.

Advanced IPv6 MPLS VPN Functionality

Advanced MPLS features such as accessing the Internet from a VPN for IPv4, multiautonomous-system backbones, and CSCs are generally the same for IPv6 as for IPv4. However, there are differences in addressing and in the way 6VPE operates over an IPv4 backbone.

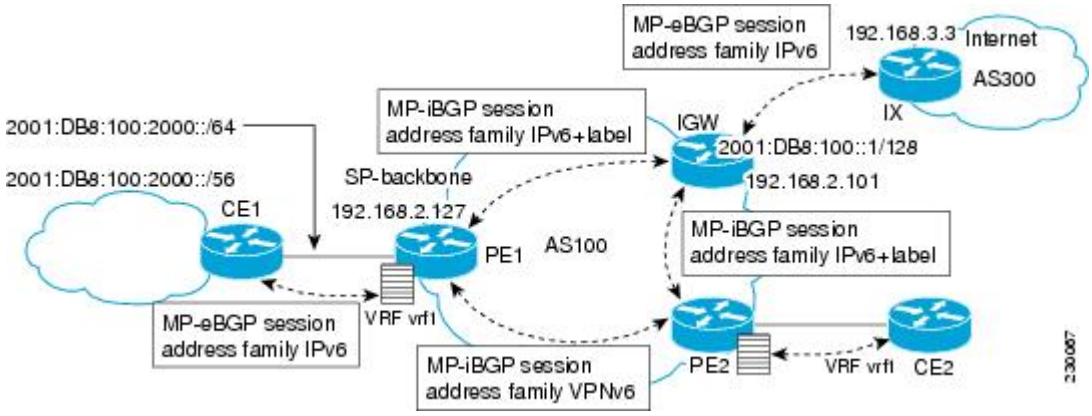
The following sections describe concepts for advanced IPv6 MPLS VPN functionality:

Internet Access

Most VPN sites require access to the Internet. RFC 4364 describes a set of models for enabling IPv4 and IPv6 VPN access to the Internet. In one model, one interface is used by the CE to connect to the Internet and a different one to connect to the VRF. Another model is in which all Internet routes are redistributed into the VRF; however, this approach has the disadvantage of requiring the Internet routes be replicated in each VRF.

In one scenario, a static route is inserted into the VRF table, with a next hop that points to the Internet gateway found in the IPv6 default table. The figure below illustrates this scenario, in which Internet access is provided to the customer in the VRF named vrf1.

Figure 29 Internet Access Topology



A customer site that has access public resources over the Internet must be known by a public prefix. Unlike IPv4, IPv6 does not offer a Network Address Translation (NAT) mechanism that translates private addresses into public addresses when leaving the site boundaries. This implies that hosts within the site speak with public addresses and appear in the public domain.

For outbound traffic, the default route configured in the VRF table at ingress PE (PE1) directs traffic for destinations outside the VPN to the Internet gateway.

For inbound traffic, a route must exist at the Internet gateway to direct the traffic for a customer site via its PE of attachment (PE1 in the figure above). This route can be distributed by the ingress PE (PE1) using multiprotocol iBGP (with the IPv6 address family configuration), so no specific configuration is needed on a per-VPN PE basis at the Internet gateway. Nevertheless, for inbound traffic at PE1, a route must exist in the default table for the customer site global prefix pointing to the VRF of the site.

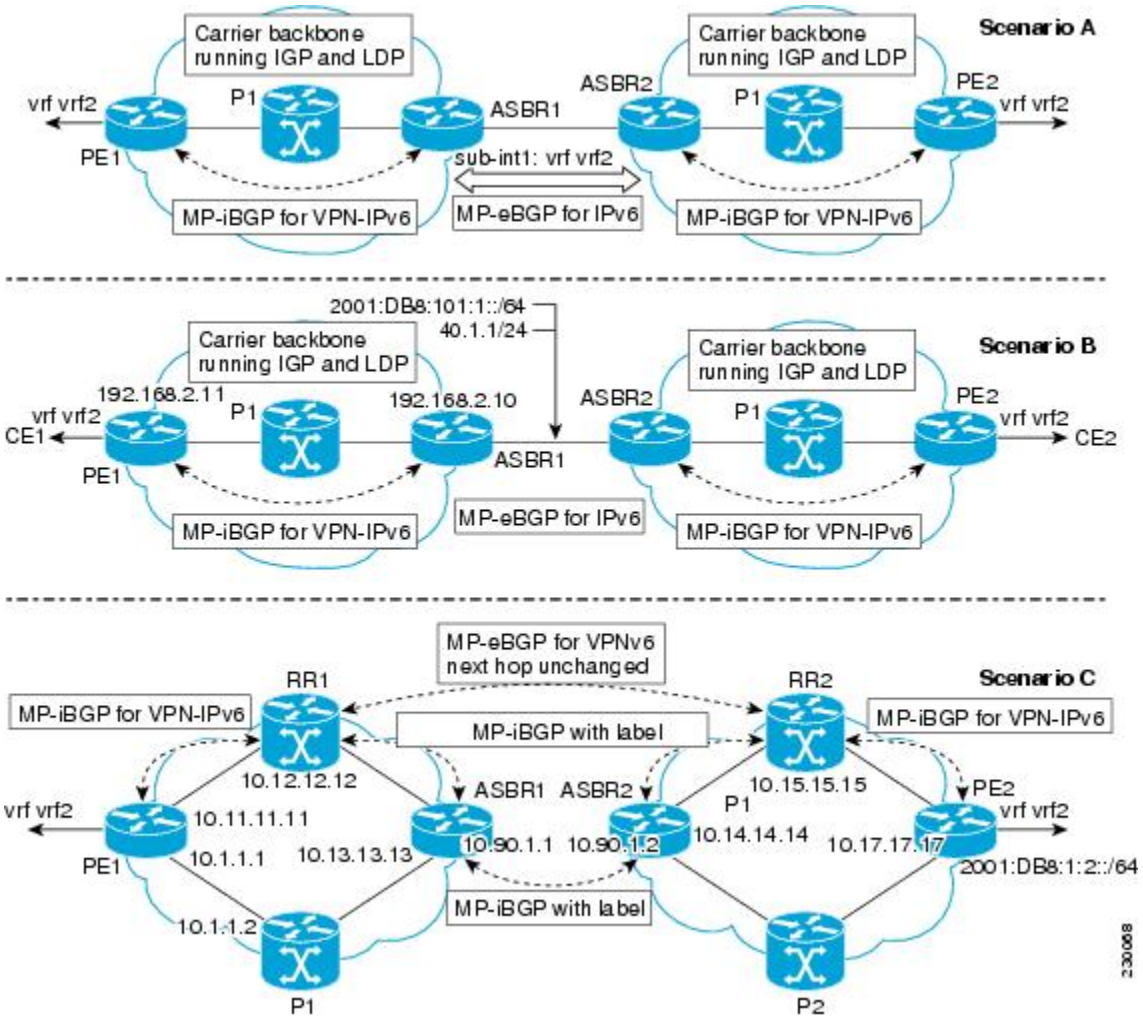
Multiautonomous-System Backbones

The problem of interprovider VPNs is similar for IPv6 and IPv4, assuming that IPv6 was deployed everywhere IPv4 was deployed.

In IPv6 deployments that cross autonomous system boundaries, providers may have to obtain a peering model, or work with the peering model put in place for VPNv4.

The figure below illustrates interprovider scenarios in IPv6 VPN.

Figure 30 Interprovider Scenarios



Depending on the network protocol used between ASBRs, the three scenarios shown in the figure above can have several implementation options. For instance, scenario B, which suggests a multiprotocol eBGP IPv6 VPN peering between ASBRs, could use either an IPv6 or an IPv4 link.

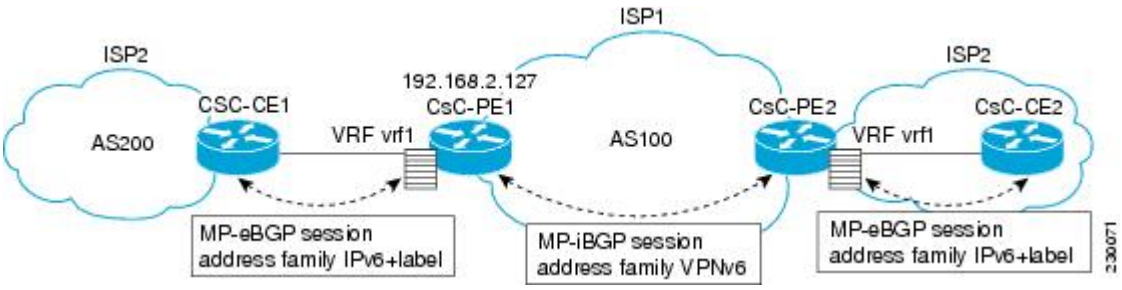
In scenario C, multihop multiprotocol eBGP redistributes IPv6 VPN routes across route reflectors in different autonomous systems. Labeled IPv4 routes to the PEs (in the 6VPE case) need to be advertised across ASBRs so that a complete labeled switch path is set up end to end.

Carrier Supporting Carriers

The CSC feature provides VPN access to a customer service provider, so this service needs to exchange routes and send traffic over the ISP MPLS backbone. The only difference from a regular PE is that it provides MPLS-to-MPLS forwarding on the CSC-CE to CSC-PE interface, rather than IP-to-MPLS forwarding.

The figure below highlights the two ISPs' interface.

Figure 31 CSC 6VPE Configuration Example



BGP IPv6 PIC Edge for IP MPLS

The BGP IPv6 PIC Edge for IP MPLS feature improves convergence for both core and edge failures after a network failure. The BGP IPv6 prefix-independent convergence (PIC) edge for IP MPLS feature creates and stores a backup or alternate path in the RIB, FIB, and in Cisco Express Forwarding, so that the backup or alternate path can immediately take over wherever a failure is detected, thus enabling fast failover.

How to Implement IPv6 VPN over MPLS

Configuring a Virtual Routing and Forwarding Instance for IPv6

A VRF is an address family-independent object that can be enabled and configured for each of the supported address families. Configuring a VRF consists of the following three steps:

- Configuring the address-family-independent part of the VRF
- Enabling and configuring IPv4 for the VRF
- Enabling and configuring IPv6 for the VRF

A VRF is given a name and an RD. The RD is configured outside the context of the address family, although the RD is used to distinguish overlapping addresses within the context of a particular BGP address family. Having separate RDs for IPv4 VPN addresses and IPv6 VPN addresses does not matter. On Cisco devices, the RDs are the same in order to simplify configuration and VPN management.

Users can configure policies in common between IPv4 and IPv6 when not using an address family context. This feature is shared route targets (import and export), and it is useful in a migration scenario, where IPv4 policies already are configured and IPv6 policies should be the same as the IPv4 policies.

The IPv4 and IPv6 address family can each be enabled and configured separately. Note that the route-target policies entered at this level override global policies that may have been specified during address family-independent configuration.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **vrf definition** *vrf-name*
4. **rd** *route-distinguisher*
5. **route-target** {**import**| **export**| **both**} *route-target-ext-community*
6. **address-family ipv4** [**mdt** | **multicast** | **tunnel** | **unicast** [**vrf** *vrf-name*] | **vrf** *vrf-name*
7. **route-target** {**import** | **export** | **both**} *route-target-ext-community*
8. **exit**
9. **address-family ipv6** [**vrf** *vrf-name*] [**unicast** | **multicast**
10. **route-target** {**import**| **export**| **both**} *route-target-ext-community*

DETAILED STEPS

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

	Command or Action	Purpose
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**vrf definition** *vrf-name* **Example:** `Device(config)# vrf definition vrf1`	Configures a VPN VRF routing table and enters VRF configuration mode.
Step 4	**rd** *route-distinguisher* **Example:** `Device(config-vrf)# rd 100:1`	Specifies the RD for a VRF.
Step 5	**route-target** {**import**| **export**| **both**} *route-target-ext-community* **Example:** `Device(config-vrf)# route target import 100:10`	Specifies the route target VPN extended communities for both IPv4 and IPv6.
Step 6	**address-family ipv4** [**mdt** | **multicast** | **tunnel** | **unicast** [**vrf** *vrf-name*] | **vrf** *vrf-name* **Example:** `Device(config)# address-family ipv4`	Enters address family configuration mode to configure a routing session using standard IPv4 address prefixes.
Step 7	**route-target** {**import** | **export** | **both**} *route-target-ext-community* **Example:** `Device(config-vrf-af)# route target import 100:11`	Specifies the route target VPN extended communities specific to IPv4.
Step 8	**exit** **Example:** `Device(config-vrf-af)# exit`	Exits address family configuration mode on this VRF.

	Command or Action	Purpose
Step 9	**address-family ipv6** [**vrf** *vrf-name*] [**unicast** | **multicast** **Example:** `Device(config-vrf)# address-family ipv6`	Enters address family configuration mode for configuring routing sessions such as BGP that use standard IPv6 address prefixes.
Step 10	**route-target** {**import**| **export**| **both**} *route-target-ext-community* **Example:** `Device(config-vrf-af)# route target import 100:12`	Specifies the route target VPN extended communities specific to IPv6.

Binding a VRF to an Interface

In order to specify which interface belongs to which VRF, use the **vrf forwarding** command for both IPv4 and IPv6. An interface cannot belong to more than one VRF. When the interface is bound to a VRF, previously configured addresses (IPv4 and IPv6) are removed, and they must be reconfigured.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type number*
4. **vrf forwarding** *vrf-name*
5. **ip address** *ip-address mask* [**secondary**
6. **ipv6 address** {*ipv6-address / prefix-length* | *prefix-name sub-bits/prefix-length*

DETAILED STEPS

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

	Command or Action	Purpose
Step 3	**interface** *type number* **Example:** `Device(config)# interface GigabitEthernet 0/0/0`	Specifies an interface type and number, and places the device in interface configuration mode.
Step 4	**vrf forwarding** *vrf-name* **Example:** `Device(config-if)# vrf forwarding vrf1`	Associates a VPN VRF with an interface or subinterface. Note that any address, IPv4 or IPv6, that was configured prior to entering this command will be removed.
Step 5	**ip address** *ip-address mask* [**secondary** **Example:** `Device(config-if)# ip address 10.10.10.1 255.255.255.0`	Configures an IPv4 address on the interface.
Step 6	**ipv6 address** {*ipv6-address* / *prefix-length* | *prefix-name sub-bits*/ *prefix-length* **Example:** `Device(config-if)# ipv6 address 2001:DB8:100:1::1/64`	Configures an IPv6 address on the interface.

Configuring a Static Route for PE-to-CE Routing

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ipv6 route** [**vrf** *vrf-name*] *ipv6-prefix* / *prefix-length* {*ipv6-address* | *interface-type interface-number* [*ipv6-address*]} [**nexthop-vrf** [*vrf-name1* | **default**]] [*administrative-distance*] [*administrative-multicast-distance* | **unicast** | **multicast**] [*next-hop-address*] [**tag** *tag*]

DETAILED STEPS

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

	Command or Action	Purpose
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**ipv6 route** [**vrf** *vrf-name*] *ipv6-prefix / prefix-length* {*ipv6-address* | *interface-type interface-number* [*ipv6-address*]} [**nexthop-vrf** [*vrf-name1* | **default**]] [*administrative-distance*] [*administrative-multicast-distance* | **unicast** | **multicast**] [*next-hop-address*] [**tag** *tag*] **Example:** `Device(config)# ipv6 route vrf vrf1 ::/0 2001:DB8:200::1 nexthop-vrf default`	Installs the specified IPv6 static route using the specified next hop.

Configuring eBGP PE-to-CE Routing Sessions

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **router bgp** *autonomous-system-number*
4. **address-family ipv6** [**vrf** *vrf-name*] [**unicast** | **multicast**]
5. **neighbor** {*ip-address* | *ipv6-address* | *peer-group-name*} **remote-as** *as-number*
6. **neighbor** { *p-address* | *peer-group-name* | ipv6-address} **activate**

DETAILED STEPS

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

	Command or Action	Purpose
Step 3	**router bgp** *autonomous-system-number* **Example:** `Device(config)# router bgp 100`	Configures the BGP routing process.
Step 4	**address-family ipv6** [**vrf** *vrf-name*] [**unicast** | **multicast**] **Example:** `Device(config-router)# address-family ipv6 vrf vrf1`	Enters address family configuration mode.
Step 5	**neighbor** {*ip-address* | *ipv6-address* | *peer-group-name*} **remote-as** *as-number* **Example:** `Device(config-router-af)# neighbor 2001:DB8:100:1::2 remote-as 200`	Adds an entry to the multiprotocol BGP neighbor table.
Step 6	**neighbor** { *p-address* | *peer-group-name* | ipv6-address} **activate** **Example:** `Device(config-router-af)# neighbor 2001:DB8:100:1::2 activate`	Enables the exchange of information for this address family with the specified BGP neighbor.

Configuring the IPv6 VPN Address Family for iBGP

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **router bgp** *autonomous-system-number*
4. **neighbor** {*ip-address* | *ipv6-address* | *peer-group-name*} **remote-as** *as-number*
5. **neighbor** {*ip-address* | *ipv6-address* | *peer-group-name*} **update-source** *interface-type interface-number*
6. **address-family vpnv6** [**unicast**]
7. **neighbor** {*ip-address* | *peer-group-name* | *ipv6-address*} **activate**
8. **neighbor** {*ip-address* | *ipv6-address* | *peer-group-name*} **send-community** [**both** | **standard** | **extended**]
9. **exit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**router bgp** *autonomous-system-number* **Example:** `Device(config)# router bgp 100`	Configures the BGP routing process.
Step 4	**neighbor** {*ip-address* | *ipv6-address* | *peer-group-name*} **remote-as** *as-number* **Example:** `Device(config-router)# neighbor 192.168.2.11 remote-as 100`	Adds an entry to the multiprotocol BGP neighbor table. • In IPv6 VPN, the peer address typically is an IPv4 address, in order to enable the BGP session to be transported over the IPv4-based core network.
Step 5	**neighbor** {*ip-address* | *ipv6-address* | *peer-group-name*} **update-source** *interface-type interface-number* **Example:** `Device(config-router)# neighbor 192.168.2.11 update-source Loopback 0`	Enables the BGP session to use a source address on the specified interface.
Step 6	**address-family vpnv6** [**unicast**] **Example:** `Device(config-router)# address-family vpnv6`	Places the device in address family configuration mode for configuring routing sessions.

	Command or Action	Purpose
Step 7	**neighbor** {*ip-address* | *peer-group-name* | *ipv6-address*} **activate** **Example:** `Device(config-router-af)# neighbor 192.168.2.11 activate`	Enables the exchange of information for this address family with the specified BGP neighbor.
Step 8	**neighbor** {*ip-address* | *ipv6-address* | *peer-group-name*} **send-community** [**both** | **standard** | **extended**] **Example:** `Device(config-router-af)# neighbor 192.168.2.11 send-community extended`	Specifies that a communities attribute should be sent to the BGP neighbor.
Step 9	**exit** **Example:** `Device(config-router-af)# exit`	Exits address family configuration mode.

Configuring Route Reflectors for Improved Scalability

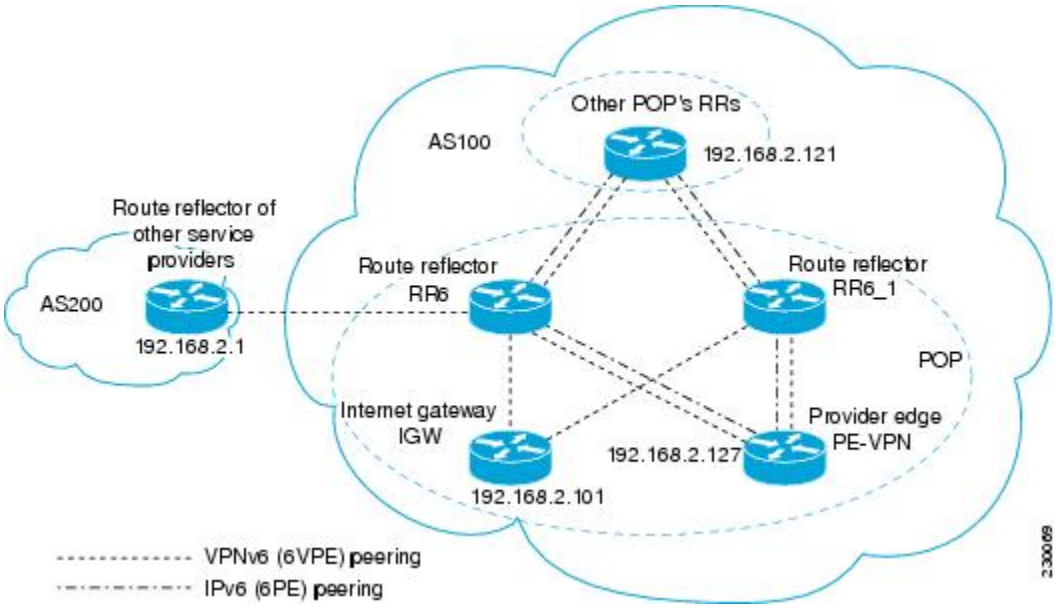
In this task, two RRs are configured for redundancy reasons. Deploying RRs improves scalability by drastically reducing the number of BGP sessions. One RR usually peers with many iBGP speakers, preventing a full mesh of BGP sessions.

In an MPLS-based core, RRs are not part of the label switch paths and can be located anywhere in the network. For example, in a flat RR design, RRs can be deployed at Level 1 points of presence (POPs) and peer together in a full-mesh topology. In a hierarchical RR design, RRs could be deployed at Level 1 and Level 2 POPs, with Level 1 POPs peering together and with Level 2 RRs.

In a typical case where 6VPE is deployed in a preexisting MPLS network (for example, providing VPNv4 services), it is likely that some RR design is already in place, and a similar RR infrastructure for IPv6 VPN

services can be deployed. The figure below illustrates the main peering points between the RR in the ISP POP and the set of its RR clients.

Figure 32 Route Reflector Peering Design



The following list of BGP RR clients must be configured at each IPv6 RR (RR6 and RR6_1 in the figure above) device, at each POP:

- PE devices (PE-VPN) of the POP providing IPv6 VPN access to the ISP customers. This includes both IPv6 VPN (6VPE) peering for interconnecting customer sites and IPv6 peering (6PE) for providing Internet access to VPN customers (see the Configuring Internet Access, page 296).
- Internet gateway (IGW) located in the POP in order to provide PE customers with access to the IPv6 Internet (see the Configuring Internet Access, page 296).
- RRs from other service providers. This feature is used to provide interautonomous-system connectivity, and it includes both IPv6 and IPv6 VPN peering. This service is described in the Configuring a Multiautonomous-System Backbone for IPv6 VPN, page 305 section.
- RRs in other POPs. All RRs peer together, with both IPv6 and IPv6 VPN address families enabled.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **router bgp** *autonomous-system-number*
4. **neighbor** {*ip-address* | *ipv6-address* | *peer-group-name*} **remote-as** *as-number*
5. **neighbor** {*ip-address* | *ipv6-address* | *peer-group-name*} **update-source** *interface-type interface-number*
6. **neighbor** {*ip-address* | *ipv6-address* | *peer-group-name*} **remote-as** *as-number*
7. **neighbor** {*ip-address* | *ipv6-address* | *peer-group-name*} **update-source** *interface-type interface-number*
8. **neighbor** {*ip-address* | *ipv6-address* | *peer-group-name*} **remote-as** *as-number*
9. **neighbor** {*ip-address* | *ipv6-address* | *peer-group-name*} **update-source** *interface-type interface-number*
10. **neighbor** {*ip-address* | *ipv6-address* | *peer-group-name*} **remote-as** *as-number*
11. **neighbor** {*ip-address* | *ipv6-address* | *peer-group-name*} **update-source** *interface-type interface-number*
12. **neighbor** {*ip-address* | *ipv6-address*| *peer-group-name*} **ebgp-multihop** [*ttl*]
13. **address-family ipv6**
14. **neighbor** {*ip-address* | *peer-group-name* | *ipv6-address*} **activate**
15. **neighbor** *ip-address* | *ipv6-address* | *peer-group-name*} **send-label**
16. **neighbor** {*ip-address* | *ipv6-address* | *peer-group-name*} **route-reflector-client**
17. **neighbor** {*ip-address* | *peer-group-name* | *ipv6-address*} **activate**
18. **neighbor** *ip-address* | *ipv6-address* | *peer-group-name*} **send-label**
19. **neighbor** {*ip-address* | *ipv6-address* | *peer-group-name*} **route-reflector-client**
20. **neighbor** {*ip-address* | *peer-group-name* | *ipv6-address*} **activate**
21. **neighbor** *ip-address* | *ipv6-address* | *peer-group-name*} **send-label**
22. **neighbor** {*ip-address* | *ipv6-address* | *peer-group-name*} **route-reflector-client**
23. **exit**
24. **address-family vpnv6** [**unicast**
25. **neighbor** {*ip-address* | *peer-group-name* | *ipv6-address*} **activate**
26. **neighbor** {*ip-address* | *ipv6-address* | *peer-group-name*} **send-community** [**both** | **standard** | **extended**]
27. **neighbor** {*ip-address* | *ipv6-address* | *peer-group-name*} **route-reflector-client**
28. **neighbor** {*ip-address* | *peer-group-name* | *ipv6-address*} **activate**
29. **neighbor** {*ip-address* | *ipv6-address* | *peer-group-name*} **send-community** [**both** | **standard** | **extended**]
30. **neighbor** {*ip-address* | *ipv6-address* | *peer-group-name*} **route-reflector-client**
31. **neighbor** {*ip-address* | *peer-group-name* | *ipv6-address*} **activate**
32. **neighbor** {*ip-address* | *ipv6-address* | *peer-group-name*} **send-community** [**both** | **standard** | **extended**]
33. **neighbor** {*ip-address* | *ipv6-address* | *peer-group-name*} **route-reflector-client**
34. **neighbor** {*ip-address* | *ipv6-address* | *peer-group-name*} **next-hop-unchanged** [**allpaths**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**router bgp** *autonomous-system-number* **Example:** `Device(config)# router bgp 100`	Configures the BGP routing process.
Step 4	**neighbor** {*ip-address* | *ipv6-address* | *peer-group-name*} **remote-as** *as-number* **Example:** `Device(config-router)# neighbor 192.168.2.101 remote-as 100`	Adds an entry to the multiprotocol BGP neighbor table, and provides peering with the Internet gateway in order to provide Internet access.
Step 5	**neighbor** {*ip-address* | *ipv6-address* | *peer-group-name*} **update-source** *interface-type interface-number* **Example:** `Device(config-router)# neighbor 192.168.2.101 update-source Loopback 0`	Enables the BGP session to use a source address on the specified interface.
Step 6	**neighbor** {*ip-address* | *ipv6-address* | *peer-group-name*} **remote-as** *as-number* **Example:** `Device(config-router)# neighbor 192.168.2.121 remote-as 100`	Adds an entry to the multiprotocol BGP neighbor table, and provides peering with the other POP's RR.

	Command or Action	Purpose
Step 7	**neighbor** {*ip-address* | *ipv6-address* | *peer-group-name*} **update-source** *interface-type interface-number* **Example:** `Device(config-router)# neighbor 192.168.2.121 update-source Loopback 0`	Enables the BGP session to use a source address on the specified interface.
Step 8	**neighbor** {*ip-address* | *ipv6-address* | *peer-group-name*} **remote-as** *as-number* **Example:** `Device(config-router)# neighbor 192.168.2.127 remote-as 100`	Adds an entry to the multiprotocol BGP neighbor table.
Step 9	**neighbor** {*ip-address* | *ipv6-address* | *peer-group-name*} **update-source** *interface-type interface-number* **Example:** `Device(config-router)# neighbor 192.168.2.127 update-source Loopback 0`	Enables the BGP session to use a source address on the specified interface.
Step 10	**neighbor** {*ip-address* | *ipv6-address* | *peer-group-name*} **remote-as** *as-number* **Example:** `Device(config-router)# neighbor 192.168.2.1 remote-as 200`	(Optional) Adds an entry to the multiprotocol BGP neighbor table, and provides peering with the RR of the peer ISP in order to provide inter-VPN service.
Step 11	**neighbor** {*ip-address* | *ipv6-address* | *peer-group-name*} **update-source** *interface-type interface-number* **Example:** `Device(config-router)# neighbor 192.168.2.1 update-source Loopback 0`	(Optional) Enables the BGP session to use a source address on the specified interface.
Step 12	**neighbor** {*ip-address* | *ipv6-address*| *peer-group-name*} **ebgp-multihop** [*ttl*] **Example:** `Device(config-router)# neighbor 192.168.2.1 ebgp-multihop`	(Optional) Accepts and attempts BGP connections to external peers residing on networks that are not directly connected.

Command or Action	Purpose
Step 13 **address-family ipv6** **Example:** `Device(config-router)# address-family ipv6`	(Optional) Enters address family configuration mode in order to provide Internet access service.
Step 14 **neighbor** {*ip-address* | *peer-group-name* | *ipv6-address*} **activate** **Example:** `Device(config-router-af)# neighbor 192.168.2.101 activate`	(Optional) Enables the exchange of information for this address family with the specified neighbor.
Step 15 **neighbor** *ip-address* | *ipv6-address* | *peer-group-name*} **send-label** **Example:** `Device(config-router-af)# neighbor 192.168.2.101 send-label`	(Optional) Enables a BGP device to send MPLS labels with BGP routes to a neighboring BGP device.
Step 16 **neighbor** {*ip-address* | *ipv6-address* | *peer-group-name*} **route-reflector-client** **Example:** `Device(config-router-af)# neighbor 192.168.2.101 route-reflector-client`	(Optional) Configures the device as a BGP route reflector and configures the specified neighbor as its client.
Step 17 **neighbor** {*ip-address* | *peer-group-name* | *ipv6-address*} **activate** **Example:** `Device(config-router-af)# neighbor 192.168.2.121 activate`	(Optional) Enables the exchange of information for this address family with the specified BGP neighbor.
Step 18 **neighbor** *ip-address* | *ipv6-address* | *peer-group-name*} **send-label** **Example:** `Device(config-router-af)# neighbor 192.168.2.121 send-label`	(Optional) Enables a BGP device to send MPLS labels with BGP routes to a neighboring BGP device.
Step 19 **neighbor** {*ip-address* | *ipv6-address* | *peer-group-name*} **route-reflector-client** **Example:** `Device(config-router-af)# neighbor 192.168.2.121 route-reflector-client`	(Optional) Configures the specified neighbor as a route reflector client.

Command or Action	Purpose
Step 20 **neighbor** {*ip-address* | *peer-group-name* | *ipv6-address*} **activate** **Example:** `Device(config-router-af)# neighbor 192.168.2.127 activate`	(Optional) Enables the exchange of information for this address family with the specified BGP neighbor.
Step 21 **neighbor** *ip-address* | *ipv6-address* | *peer-group-name*} **send-label** **Example:** `Device(config-router-af)# neighbor 192.168.2.127 send-label`	(Optional) Enables a BGP device to send MPLS labels with BGP routes to a neighboring BGP device.
Step 22 **neighbor** {*ip-address* | *ipv6-address* | *peer-group-name*} **route-reflector-client** **Example:** `Device(config-router-af)# neighbor 192.168.2.127 route-reflector-client`	(Optional) Configures the specified neighbor as a route reflector client.
Step 23 **exit** **Example:** `Device(config-router-af)# exit`	(Optional) Exits address family configuration mode.
Step 24 **address-family vpnv6** [**unicast** **Example:** `Device(config-router)# address-family vpnv6`	Places the device in address family configuration mode for configuring routing sessions.
Step 25 **neighbor** {*ip-address* | *peer-group-name* | *ipv6-address*} **activate** **Example:** `Device(config-router-af)# neighbor 192.168.2.121 activate`	Enables the exchange of information for this address family with the specified BGP neighbor.
Step 26 **neighbor** {*ip-address* | *ipv6-address* | *peer-group-name*} **send-community** [**both** | **standard** | **extended**] **Example:** `Device(config-router-af)# neighbor 192.168.2.21 send-community extended`	Specifies that a communities attribute should be sent to the BGP neighbor.

	Command or Action	Purpose
Step 27	**neighbor** {*ip-address* | *ipv6-address* | *peer-group-name*} **route-reflector-client** **Example:** `Device(config-router-af)# neighbor 192.168.2.121 route-reflector-client`	Configures the specified neighbor as a route reflector client.
Step 28	**neighbor** {*ip-address* | *peer-group-name* | *ipv6-address*} **activate** **Example:** `Device(config-router-af)# neighbor 192.168.2.127 activate`	Enables the exchange of information for this address family with the specified BGP neighbor.
Step 29	**neighbor** {*ip-address* | *ipv6-address* | *peer-group-name*} **send-community** [**both** | **standard** | **extended**] **Example:** `Device(config-router-af)# neighbor 192.168.2.127 send-community extended`	Specifies that a communities attribute should be sent to the BGP neighbor.
Step 30	**neighbor** {*ip-address* | *ipv6-address* | *peer-group-name*} **route-reflector-client** **Example:** `Device(config-router-af)# neighbor 192.168.2.127 route-reflector-client`	Configures the specified neighbor as a route reflector client.
Step 31	**neighbor** {*ip-address* | *peer-group-name* | *ipv6-address*} **activate** **Example:** `Device(config-router-af)# neighbor 192.168.2.1 activate`	Enables the exchange of information for this address family with the specified BGP neighbor.
Step 32	**neighbor** {*ip-address* | *ipv6-address* | *peer-group-name*} **send-community** [**both** | **standard** | **extended**] **Example:** `Device(config-router-af)# neighbor 192.168.2.1 send-community extended`	Specifies that a communities attribute should be sent to the BGP neighbor.

	Command or Action	Purpose
Step 33	**neighbor** {*ip-address* | *ipv6-address* | *peer-group-name*} **route-reflector-client** **Example:** `Device(config-router-af)# neighbor 192.168.2.1 route-reflector-client`	Configures the specified neighbor as a route reflector client.
Step 34	**neighbor** {*ip-address* | *ipv6-address* | *peer-group-name*} **next-hop-unchanged** [**allpaths** **Example:** `Device(config-router-af)# neighbor 192.168.2.1 next-hop-unchanged allpaths`	Enables an EBGP multihop peer to propagate to the next hop unchanged for paths.

Configuring Internet Access

Customers with IPv6 VPN access need to have access to the Internet through IPv6. The design of this service is similar to a global Internet access service. 6VPE devices located in a Level 1 POP (colocated with an IGW device) can access the IGW natively, whereas 6VPE devices located in Level 2 and Level 3 POPs with no direct access to the IGW can access the IGW in their closest Level 1 POP over 6PE.

Configuring VPN Internet access in such a 6VPE device involves configuring BGP peering with the IGW (in most cases through the IPv6 RR, as described in the Configuring Route Reflectors for Improved Scalability section). Then the user must configure cross-table routing to enable communication between the private domain (the VRF) and the public domain (the Internet).

The figure above illustrates the following configuration tasks:

- Configuring the Internet Gateway, page 296
- Configuring the IPv6 VPN PE, page 301

Configuring the Internet Gateway

- Configuring iBGP 6PE Peering to the VPN PE, page 296
- Configuring the Internet Gateway as the Gateway to the Public Domain, page 298
- Configuring eBGP Peering to the Internet, page 299

Configuring iBGP 6PE Peering to the VPN PE

Perform this task to configure iBGP 6PE peering in the VPN PE.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **router bgp** *autonomous-system-number*
4. **neighbor** {*ip-address* | *ipv6-address* | *peer-group-name*} **remote-as** *as-number*
5. **neighbor** {*ip-address* | *ipv6-address* | *peer-group-name*} **update-source** *interface-type interface-number*
6. **address-family ipv6**
7. **neighbor** {*ip-address* | *peer-group-name* | *ipv6-address*} **activate**
8. **neighbor** *ip-address* | *ipv6-address* | *peer-group-name*} **send-label**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**router bgp** *autonomous-system-number* **Example:** `Device(config)# router bgp 100`	Configures the BGP routing process.
Step 4	**neighbor** {*ip-address* | *ipv6-address* | *peer-group-name*} **remote-as** *as-number* **Example:** `Device(config-router)# neighbor 192.168.2.127 remote-as 100`	Adds an entry to the multiprotocol BGP neighbor table to provide peering with the VPN PE.

	Command or Action	Purpose
Step 5	**neighbor** {*ip-address* | *ipv6-address* | *peer-group-name*} **update-source** *interface-type interface-number* **Example:** `Device(config-router)# neighbor 192.168.2.127 update-source Loopback 0`	Enables the BGP session to use a source address on the specified interface.
Step 6	**address-family ipv6** **Example:** `Device(config-router)# address-family ipv6`	Enters address family configuration mode in order to exchange global table reachability.
Step 7	**neighbor** {*ip-address* | *peer-group-name* | *ipv6-address*} **activate** **Example:** `Device(config-router-af)# neighbor 192.168.2.127 activate`	Enables the exchange of information for this address family with the specified BGP neighbor.
Step 8	**neighbor** *ip-address* | *ipv6-address* | *peer-group-name*} **send-label** **Example:** `Device(config-router-af)# neighbor 192.168.2.127 send-label`	Enables a BGP device to send MPLS labels with BGP routes to a neighboring BGP device, and allows the PE VPN to reach the Internet gateway over MPLS.

Configuring the Internet Gateway as the Gateway to the Public Domain

Use the 6PE peering configuration established in the Configuring iBGP 6PE Peering to the VPN PE, page 296.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **router bgp** *autonomous-system-number*
4. **address-family ipv6**
5. **network** *ipv6-address* / *prefix-length*
6. **exit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**router bgp** *autonomous-system-number* **Example:** `Device(config)# router bgp 100`	Configures the BGP routing process.
Step 4	**address-family ipv6** **Example:** `Device(config-router)# address-family ipv6`	Enters address family configuration mode in order to exchange global table reachability.
Step 5	**network** *ipv6-address* / *prefix-length* **Example:** `Device(config-router-af)# network 2001:DB8:100::1/128`	Configures the network source of the next hop to be used by the PE VPN.
Step 6	**exit** **Example:** `Device(config-router-af)# exit`	Exits address family configuration mode.

Configuring eBGP Peering to the Internet

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **router bgp** *autonomous-system-number*
4. **neighbor** {*ip-address* | *ipv6-address* | *peer-group-name*} **remote-as** *as-number*
5. **address-family ipv6**
6. **neighbor** {*ip-address* | *peer-group-name* | *ipv6-address*} **activate**
7. **aggregate-address** *address mask* [**as-set**] [**summary-only**] [**suppress-map** *map-name*] [**advertise-map** *map-name*] [**attribute-map** *map-name*]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**router bgp** *autonomous-system-number* **Example:** `Device(config)# router bgp 100`	Configures the BGP routing process.
Step 4	**neighbor** {*ip-address* | *ipv6-address* | *peer-group-name*} **remote-as** *as-number* **Example:** `Device(config-router)# neighbor FE80::300::1 GigabitEthernet0/0/0 remote-as 300`	Adds an entry to the multiprotocol BGP neighbor table, and provides peering with PE (PE-VPN). • Note that the peering is done over link-local addresses.
Step 5	**address-family ipv6** **Example:** `Device(config-router)# address-family ipv6`	Enters address family configuration mode in order to exchange global table reachability.

	Command or Action	Purpose
Step 6	**neighbor** {*ip-address* | *peer-group-name* | *ipv6-address*} **activate** **Example:** `Device(config-router-af)# neighbor FE80::300::1 GigabitEthernet0/0/0 activate`	Enables the exchange of information for this address family with the specified BGP neighbor.
Step 7	**aggregate-address** *address mask* [**as-set**] [**summary-only**] [**suppress-map** *map-name*] [**advertise-map** *map-name*] [**attribute-map** *map-name*] **Example:** `Device(config-router-af)# aggregate-address 2001:DB8::/32 summary-only`	Creates an aggregate prefix before advertising it to the Internet.

Configuring the IPv6 VPN PE

Configuring a Default Static Route from the VRF to the Internet Gateway

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ipv6 route** [**vrf** *vrf-name*] *ipv6-prefix*/*prefix-length* {*ipv6-address* | *interface-type interface-number* [*ipv6-address*]} [**nexthop-vrf** [*vrf-name1* | **default**]] [*administrative-distance*] [*administrative-multicast-distance* | **unicast** | **multicast**] [*next-hop-address*] [**tag** *tag*]

DETAILED STEPS

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

Command or Action	Purpose
Step 2 **configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3 **ipv6 route** [**vrf** *vrf-name*] *ipv6-prefix*/*prefix-length* {*ipv6-address* | *interface-type interface-number* [*ipv6-address*]} [**nexthop-vrf** [*vrf-name1* | **default**]] [*administrative-distance*] [*administrative-multicast-distance* | **unicast** | **multicast**] [*next-hop-address*] [**tag** *tag*] **Example:** `Device(config)# ipv6 route vrf vrf1 ::/0 2001:DB8:100::1 nexthop-vrf default`	Configures a default static route from the VRF to the Internet gateway to allow outbound traffic to leave the VRF.

Configuring a Static Route from the Default Table to the VRF

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ipv6 route** [**vrf** *vrf-name*] *ipv6-prefix*/*prefix-length* {*ipv6-address* | *interface-type interface-number* [*ipv6-address*]} [**nexthop-vrf** [*vrf-name1* | **default**]] [*administrative-distance*] [*administrative-multicast-distance* | **unicast** | **multicast**] [*next-hop-address*] [**tag** *tag*]

DETAILED STEPS

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

Command or Action	Purpose
Step 3 **ipv6 route** [**vrf** *vrf-name*] *ipv6-prefix*/*prefix-length* {*ipv6-address* | *interface-type interface-number* [*ipv6-address*]} [**nexthop-vrf** [*vrf-name1* | **default**]] [*administrative-distance*] [*administrative-multicast-distance* | **unicast** | **multicast**] [*next-hop-address*] [**tag** *tag*] **Example:** `Device(config)# ipv6 route 2001:DB8:100:2000::/64 nexthop-vrf vrf1`	Configures a static route from the default table to the VRF to allow inbound traffic to reach the VRF.

Configuring iBGP 6PE Peering to the Internet Gateway

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **router bgp** *autonomous-system-number*
4. **neighbor** {*ip-address* | *ipv6-address* | *peer-group-name*} **remote-as** *as-number*
5. **neighbor** {*ip-address* | *ipv6-address* | *peer-group-name*} **update-source** *interface-type interface-number*
6. **address-family ipv6** [**vrf** *vrf-name*] [**unicast** | **multicast**]
7. **neighbor** {*ip-address* | *peer-group-name* | *ipv6-address*} **activate**
8. **neighbor** { *ip-address* | *ipv6-address* | *peer-group-name*} **send-label**
9. **network** *ipv6-address* / *prefix-length*

DETAILED STEPS

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

Command or Action	Purpose
Step 3 **router bgp** *autonomous-system-number* **Example:** `Device(config)# router bgp 100`	Configures the BGP routing process.
Step 4 **neighbor** {*ip-address* | *ipv6-address* | *peer-group-name*} **remote-as** *as-number* **Example:** `Device(config-router)# neighbor 192.168.2.101 remote-as 100`	Adds an entry to the multiprotocol BGP neighbor table for peering with the Internet gateway.
Step 5 **neighbor** {*ip-address* | *ipv6-address* | *peer-group-name*} **update-source** *interface-type interface-number* **Example:** `Device(config-router)# neighbor 192.168.2.101 update-source Loopback 0`	Enables the BGP session to use a source address on the specified interface.
Step 6 **address-family ipv6** [**vrf** *vrf-name*] [**unicast** | **multicast**] **Example:** `Device(config-router)# address-family ipv6`	Enters address family configuration mode to exchange global table reachability.
Step 7 **neighbor** {*ip-address* | *peer-group-name* | *ipv6-address*} **activate** **Example:** `Device(config-router-af)# neighbor 192.168.2.101 activate`	Enables the exchange of information for this address family with the specified BGP neighbor.
Step 8 **neighbor** { *ip-address* | *ipv6-address* | *peer-group-name*} **send-label** **Example:** `Device(config-router-af)# neighbor 192.168.2.101 send-label`	Enables label exchange for this address family to this neighbor to enable the VPN PE to reach the Internet gateway over MPLS.
Step 9 **network** *ipv6-address* / *prefix-length* **Example:** `Device(config-router-af)# network 2001:DB8:100:2000::/64`	Provides the VRF prefix to the Internet gateway.

Configuring a Multiautonomous-System Backbone for IPv6 VPN

Two VPN sites may be connected to different autonomous systems because the sites are connected to different service providers. The PE devices attached to that VPN is then unable to maintain iBGP connections with each other or with a common route reflector. In this situation, there must be some way to use eBGP to distribute VPN-IPv6 addresses.

The following configuration example illustrates two scenarios, one in which a multiprotocol eBGP-IPv6 VPN peering between ASBRs uses an IPv4 link, and the same scenario using an IPv6 link. If the peering between ASBRs is performed over an IPv4 link, the BGP configuration on ASBR1 is as follows:

```
router bgp 1001
no bgp default ipv4-unicast
no bgp default route-target filter
neighbor 192.1.1.1 remote-as 1002
neighbor 192.168.2.11 remote-as 1001
neighbor 192.168.2.11 update-source Loopback1
!
 address-family vpnv6
!Peering to ASBR2 over an IPv4 link
 neighbor 192.1.1.1 activate
 neighbor 192.1.1.1 send-community extended
!Peering to PE1 over an IPv4 link
 neighbor 192.168.2.11 activate
 neighbor 192.168.2.11 next-hop-self
 neighbor 192.168.2.11 send-community extended
```

If the peering between ASBRs is performed over an IPv6 link, the BGP configuration on ASBR1 is as follows:

```
router bgp 1001
neighbor 2001:DB8:101::72d remote-as 1002
!
 address-family vpnv6
!Peering to ASBR2 over an IPv6 link
 neighbor 2001:DB8:101::72d activate
 neighbor 2001:DB8:101::72d send-community extended
```

The next several tasks describe how to configure the PE VPN for a multiautonomous-system backbone using multihop multiprotocol eBGP to redistribute VPN routes across RRs in different autonomous systems. Labeled IPv4 routes to the PEs are advertised across ASBRs so that a complete label switch path (LSP) is set up end to end.

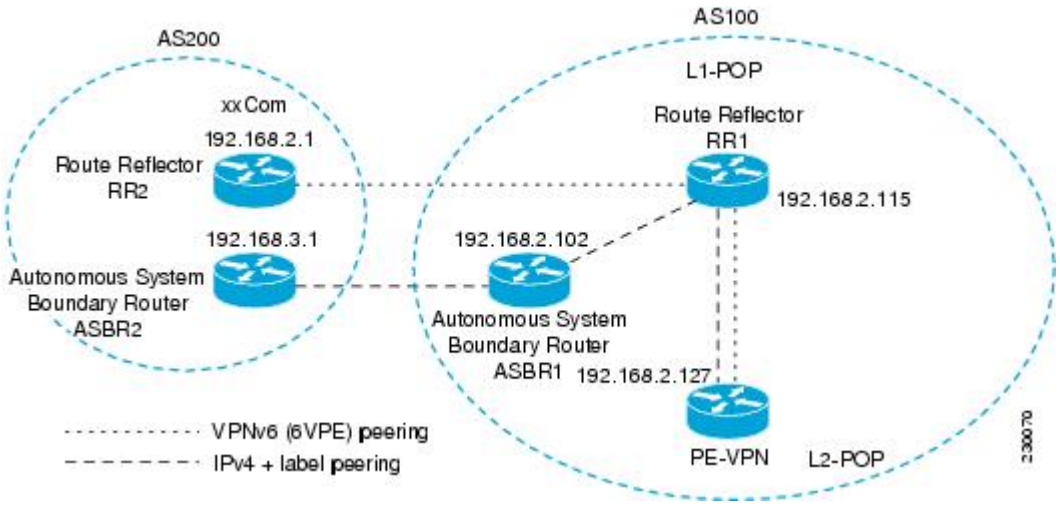
In this scenario, the ASBRs are not VPN aware; only the RRs are VPN aware. The following configuration should be available and understood:

- The ASBRs are providing the PEs' loopback addresses to service providers they peer with. That includes:
  - The VPN PE's IPv4 loopback address (/32) for enabling next-hop resolution at the remote service provider location.
  - The VPN RR's IPv4 loopback address (/32) for enabling interprovider (inter-RR) eBGP peering.
- For the VPN PE's IPv4 loopback address, the address providing is performed over multiprotocol BGP, with the label, up to the remote PEs, so that the label establishes an end-to-end LSP. Therefore, the following MP-BGP peering was set up for VPNv4:
  - VPN PEs are iBGP peering with VPN RRs.
  - ASBRs are iBGP peering with VPN RRs.
  - ASBRs are eBGP peering with the remote service provider ASBR.
- The VPN RRs of each service provider are peering together over eBGP and exchanging VPN routes. The next hop is forwarded unchanged, so that the end-to-end LSP is not via RRs.

To enable IPv6 VPN interautonomous-system access in this scenario, the ISP needs to modify the configurations at the PE VPN and at the RR. The same RRs are set up to provide a similar service for VPNv4. In that context, because the peering between the RR and the ASBR and between ASBRs is solely to exchange labels for IPv4 next hops used by both IPv4 VPN and IPv6 VPN, the ASBRs remain completely IPv6 unaware, and no configuration change is required there.

The figure below shows the BGP peering points required to enable IPv6 interprovider connectivity from the PE-VPN device (providing IPv6 VPN access) to the xxCom network.

Figure 33 BGP Peering Points for Enabling Interautonomous System Scenario C



The following additional BGP peerings are necessary to enable interautonomous-system communication from the IPv6 VPN PE located in the Level 2 POP:

- IPv4 with label peering from the PE VPN to the route reflector named RR1 (which is already configured if VPNv4 interautonomous system is deployed on the same nodes, using the same LSP).
- IPv4 with label peering from RR1 to ASBR1.
- IPv4 with label peering between ASBR1 and ASBR2.
- IPv6 VPN peering between RR1 and RR2 (which is the route reflector in the other autonomous systems) to exchange IPv6 VPN routes.
- IPv6 VPN peering with RR1. If the same route reflectors used to scale the IPv6 VPN service are used for interautonomous-system capability, then this function might also be already configured (see the Configuring Route Reflectors for Improved Scalability, page 288).

Configuring the multiautonomous-system backbone for IPv6 VPN consists of the following tasks:

- Configuring the PE VPN for a Multiautonomous-System Backbone, page 306
- Configuring the Route Reflector for a Multiautonomous-System Backbone, page 309
- Configuring the ASBR, page 319

Configuring the PE VPN for a Multiautonomous-System Backbone

- Configuring iBGP IPv6 VPN Peering to a Route Reflector, page 307
- Configuring IPv4 and Label iBGP Peering to a Route Reflector, page 308

Configuring iBGP IPv6 VPN Peering to a Route Reflector

Perform this task to configure iBGP IPv6 VPN peering to a route reflector named RR1.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **router bgp** *autonomous-system-number*
4. **neighbor** {*ip-address* | *ipv6-address* | *peer-group-name*} **remote-as** *as-number*
5. **neighbor** {*ip-address* | *ipv6-address* | *peer-group-name*} **update-source** *interface-type interface-number*
6. **address-family vpnv6** [**unicast**]
7. **neighbor** {*ip-address* | *peer-group-name* | *ipv6-address*} **activate**
8. **neighbor** {*ip-address* | *ipv6-address* | **peer-group-name**} **send-community** [**both** | **standard** | **extended**]
9. **exit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**router bgp** *autonomous-system-number* **Example:** `Device(config)# router bgp 100`	Configures the BGP routing process.
Step 4	**neighbor** {*ip-address* | *ipv6-address* | *peer-group-name*} **remote-as** *as-number* **Example:** `Device(config-router)# neighbor 192.168.2.115 remote-as 100`	Adds an entry to the multiprotocol BGP neighbor table for peering with the route reflector with interautonomous-system functionality.

Command or Action	Purpose
Step 5 **neighbor** {*ip-address* | *ipv6-address* | *peer-group-name*} **update-source** *interface-type interface-number* **Example:** `Device(config-router)# neighbor 192.168.2.115 update-source Loopback 0`	Enables the BGP session to use a source address on the specified interface.
Step 6 **address-family vpnv6** [**unicast**] **Example:** `Device(config-router)# address-family vpnv6`	(Optional) Places the device in address family configuration mode for configuring routing sessions.
Step 7 **neighbor** {*ip-address* | *peer-group-name* | *ipv6-address*} **activate** **Example:** `Device(config-router-af)# neighbor 192.168.2.115 activate`	Enables the exchange of information for this address family with the specified BGP neighbor.
Step 8 **neighbor** {*ip-address* | *ipv6-address* | **peer-group-name**} **send-community** [**both** | **standard** | **extended**] **Example:** `Device(config-router-af)# neighbor 192.168.2.115 send-community extended`	Specifies that a communities attribute should be sent to the BGP neighbor.
Step 9 **exit** **Example:** `Device(config-router-af)# exit`	Exits address family configuration mode.

Configuring IPv4 and Label iBGP Peering to a Route Reflector

Perform this task to configure IPv4 and label iBGP peering to a route reflector named RR1.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **router bgp** *autonomous-system-number*
4. **address-family ipv4** [**mdt** | **multicast** | **tunnel** | **unicast** [**vrf** *vrf-name*] | **vrf** *vrf-name*]
5. **neighbor** {*ip-address* | *peer-group-name* | *ipv6-address*} **activate**
6. **neighbor** { *ip-address* | *ipv6-address* | *peer-group-name*} **send-label**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**router bgp** *autonomous-system-number* **Example:** `Device(config)# router bgp 100`	Configures the BGP routing process.
Step 4	**address-family ipv4** [**mdt** | **multicast** | **tunnel** | **unicast** [**vrf** *vrf-name*] | **vrf** *vrf-name*] **Example:** `Device(config-router)# address-family ipv4`	Enters address family configuration mode to configure a routing session using standard IPv4 address prefixes.
Step 5	**neighbor** {*ip-address* | *peer-group-name* | *ipv6-address*} **activate** **Example:** `Device(config-router-af)# neighbor 192.168.2.115 activate`	Enables the exchange of information for this address family with the specified BGP neighbor.
Step 6	**neighbor** { *ip-address* | *ipv6-address* | *peer-group-name*} **send-label** **Example:** `Device(config-router-af)# neighbor 192.168.2.115 send-label`	Enables label exchange for this address family to this neighbor in order to receive remote PE peer IPv4 loopback with label via RR1 in order to set up an end-to-end LSP.

Configuring the Route Reflector for a Multiautonomous-System Backbone

• Configuring Peering to the PE VPN, page 310

- Configuring the Route Reflector, page 312
- Configuring Peering to the Autonomous System Boundary Router, page 315
- Configuring Peering to Another ISP Route Reflector, page 317

Configuring Peering to the PE VPN

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **router bgp** *autonomous-system-number*
4. **neighbor** {*ip-address* | *ipv6-address* | *peer-group-name*} **remote-as** *as-number*
5. **neighbor** {*ip-address* | *ipv6-address* | *peer-group-name*} **update-source** *interface-type interface-number*
6. **address-family vpnv6** [**unicast**]
7. **neighbor** {*ip-address* | *peer-group-name* | *ipv6-address*} **activate**
8. **neighbor** {*ip-address* | *ipv6-address* | *peer-group-name*} **send-community** [**both** | **standard** | **extended**]
9. **exit**
10. **address-family ipv4** [**mdt** | **multicast** | **tunnel** | **unicast** [**vrf** *vrf-name*] | **vrf** *vrf-name*]
11. **neighbor** {*ip-address* | *peer-group-name* | *ipv6-address*} **activate**
12. **neighbor** *ip-address* | *ipv6-address* | *peer-group-name*} **send-label**
13. **exit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**router bgp** *autonomous-system-number* **Example:** `Device(config)# router bgp 100`	Configures the BGP routing process.

	Command or Action	Purpose
Step 4	**neighbor** {*ip-address* | *ipv6-address* | *peer-group-name*} **remote-as** *as-number* **Example:** `Device(config-router)# neighbor 192.168.2.115 remote-as 100`	Adds an entry to the multiprotocol BGP neighbor table for peering with the route reflector for InterAS.
Step 5	**neighbor** {*ip-address* | *ipv6-address* | *peer-group-name*} **update-source** *interface-type interface-number* **Example:** `Device(config-router)# neighbor 192.168.2.115 update-source Loopback 0`	Enables the BGP session to use a source address on the specified interface.
Step 6	**address-family vpnv6** [**unicast**] **Example:** `Device(config-router)# address-family vpnv6`	(Optional) Places the device in address family configuration mode.
Step 7	**neighbor** {*ip-address* | *peer-group-name* | *ipv6-address*} **activate** **Example:** `Device(config-router-af)# neighbor 192.168.2.115 activate`	Enables the exchange of information for this address family with the specified BGP neighbor.
Step 8	**neighbor** {*ip-address* | *ipv6-address* | *peer-group-name*} **send-community** [**both** | **standard** | **extended**] **Example:** `Device(config-router-af)# neighbor 192.168.2.115 send-community extended`	Specifies that a community attribute should be sent to the BGP neighbor.
Step 9	**exit** **Example:** `Device(config-router-af)# exit`	Exits address family configuration mode.

	Command or Action	Purpose
Step 10	**address-family ipv4** [**mdt** | **multicast** | **tunnel** | **unicast** [**vrf** *vrf-name*] | **vrf** *vrf-name*] **Example:** `Device(config-router)# address-family ipv4`	Enters address family configuration mode to configure a routing session using standard IPv4 address prefixes.
Step 11	**neighbor** {*ip-address* | *peer-group-name* | *ipv6-address*} **activate** **Example:** `Device(config-router-af)# neighbor 192.168.2.115 activate`	Enables the exchange of information for this address family with the specified BGP neighbor.
Step 12	**neighbor** *ip-address* | *ipv6-address* | *peer-group-name*} **send-label** **Example:** `Device(config-router-af)# neighbor 192.168.2.115 send-label`	Enables label exchange for this address family to this neighbor in order to send to the local PE the remote PE IPv4 loopback with a label in order to set up an end-to-end LSP.
Step 13	**exit** **Example:** `Device(config-router-af)# exit`	Exits address family configuration mode.

Configuring the Route Reflector

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **router bgp** *autonomous-system-number*
4. **neighbor** {*ip-address* | *ipv6-address* | *peer-group-name*} **remote-as** *as-number*
5. **neighbor** {*ip-address* | *ipv6-address* | *peer-group-name*} **update-source** *interface-type interface-number*
6. **address-family vpnv6** [**unicast**]
7. **neighbor** {*ip-address* | *peer-group-name* | *ipv6-address*} **activate**
8. **neighbor** {*ip-address* | *ipv6-address* | *peer-group-name*} **send-community** [**both** | **standard** | **extended**]
9. **neighbor** *ip-address* | *ipv6-address* | *peer-group-name*} **route-reflector-client**
10. **exit**
11. **address-family ipv4** [**mdt** | **multicast** | **tunnel** | **unicast** [**vrf** *vrf-name*] | **vrf** *vrf-name*]
12. **neighbor** {*ip-address* | *peer-group-name* | *ipv6-address*} **activate**
13. **neighbor** *ip-address* | *ipv6-address* | *peer-group-name*} **send-label**
14. **exit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**router bgp** *autonomous-system-number* **Example:** `Device(config)# router bgp 100`	Configures the BGP routing process.

	Command or Action	Purpose
Step 4	**neighbor** {*ip-address* | *ipv6-address* | *peer-group-name*} **remote-as** *as-number* **Example:** `Device(config-router)# neighbor 192.168.2.127 remote-as 100`	Adds an entry to the multiprotocol BGP neighbor table for peering with the VPN PE for InterAS.
Step 5	**neighbor** {*ip-address* | *ipv6-address* | *peer-group-name*} **update-source** *interface-type interface-number* **Example:** `Device(config-router)# neighbor 192.168.2.127 update-source Loopback 0`	Enables the BGP session to use a source address on the specified interface.
Step 6	**address-family vpnv6** [**unicast**] **Example:** `Device(config-router)# address-family vpnv6`	(Optional) Places the device in address family configuration mode.
Step 7	**neighbor** {*ip-address* | *peer-group-name* | *ipv6-address*} **activate** **Example:** `Device(config-router-af)# neighbor 192.168.2.127 activate`	Enables the exchange of information for this address family with the specified neighbor.
Step 8	**neighbor** {*ip-address* | *ipv6-address* | *peer-group-name*} **send-community** [**both** | **standard** | **extended**] **Example:** `Device(config-router-af)# neighbor 192.168.2.127 send-community extended`	Specifies that a community attribute should be sent to the BGP neighbor.
Step 9	**neighbor** *ip-address* | *ipv6-address* | *peer-group-name*} **route-reflector-client** **Example:** `Device(config-router-af)# neighbor 192.168.2.127 route-reflector-client`	Configures the specified neighbor as a route reflector client.

	Command or Action	Purpose
Step 10	**exit** **Example:** `Device(config-router-af)# exit`	Exits address family configuration mode.
Step 11	**address-family ipv4** [**mdt** | **multicast** | **tunnel** | **unicast** [**vrf** *vrf-name*] | **vrf** *vrf-name*] **Example:** `Device(config-router)# address-family ipv4`	Enters address family configuration mode to configure a routing session using standard IPv4 address prefixes.
Step 12	**neighbor** {*ip-address* | *peer-group-name* | *ipv6-address*} **activate** **Example:** `Device(config-router-af)# neighbor 192.168.2.127 activate`	Enables the exchange of information for this address family with the specified neighbor.
Step 13	**neighbor** *ip-address* | *ipv6-address* | *peer-group-name*} **send-label** **Example:** `Device(config-router-af)# neighbor 192.168.2.127 send-label`	Enables label exchange for this address family to this neighbor in order to send to the local PE the remote PE IPv4 loopback with a label in order to set up an end-to-end LSP.
Step 14	**exit** **Example:** `Device(config-router-af)# exit`	Exits address family configuration mode.

Configuring Peering to the Autonomous System Boundary Router

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **router bgp** *autonomous-system-number*
4. **neighbor** {*ip-address* | *ipv6-address* | *peer-group-name*} **remote-as** *as-number*
5. **neighbor** {*ip-address* | *ipv6-address* | *peer-group-name*} **update-source** *interface-type interface-number*
6. **address-family ipv4** [**mdt** | **multicast** | **tunnel** | **unicast** [**vrf** *vrf-name*] | **vrf** *vrf-name*]
7. **neighbor** {*ip-address* | *peer-group-name* | *ipv6-address*} **activate**
8. **neighbor** { *ip-address* | *ipv6-address* | *peer-group-name*} **send-label**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**router bgp** *autonomous-system-number* **Example:** `Device(config)# router bgp 100`	Configures the BGP routing process.
Step 4	**neighbor** {*ip-address* | *ipv6-address* | *peer-group-name*} **remote-as** *as-number* **Example:** `Device(config-router)# neighbor 192.168.2.102 remote-as 100`	Adds an entry to the multiprotocol BGP neighbor table for peering with the ASBR1.

Command or Action	Purpose
Step 5 **neighbor** {*ip-address* | *ipv6-address* | *peer-group-name*} **update-source** *interface-type interface-number* **Example:** `Device(config-router)# neighbor 192.168.2.102 update-source Loopback 0`	Enables the BGP session to use a source address on the specified interface.
Step 6 **address-family ipv4** [**mdt** | **multicast** | **tunnel** | **unicast** [**vrf** *vrf-name*] | **vrf** *vrf-name*] **Example:** `Device(config-router)# address-family ipv4`	Enters address family configuration mode to configure a routing session using standard IPv4 address prefixes.
Step 7 **neighbor** {*ip-address* | *peer-group-name* | *ipv6-address*} **activate** **Example:** `Device(config-router-af)# neighbor 192.168.2.102 activate`	Enables the exchange of information for this address family with the specified BGP neighbor.
Step 8 **neighbor** { *ip-address* | *ipv6-address* | *peer-group-name*} **send-label** **Example:** `Device(config-router-af)# neighbor 192.168.2.102 send-label`	Enables label exchange for this address family to this neighbor in order to receive the remote PE IPv4 loopback with the label set to an end-to-end LSP.

Configuring Peering to Another ISP Route Reflector

Perform this task to configure peering to an ISP route reflector named RR2.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **router bgp** *autonomous-system-number*
4. **neighbor** {*ip-address* | *ipv6-address* | *peer-group-name*} **remote-as** *as-number*
5. **neighbor** {*ip-address* | *ipv6-address* | *peer-group-name*} **update-source** *interface-type interface-number*
6. **neighbor** {*ip-address* | *ipv6-address*| *peer-group-name*} **ebgp-multihop** [*ttl*]
7. **address-family vpnv6** [**unicast**]
8. **neighbor** {*ip-address* | *peer-group-name* | *ipv6-address*} **activate**
9. **neighbor** {*ip-address* | *ipv6-address* | **peer-group-name**} **send-community** [**both** | **standard** | **extended**]
10. **neighbor** {*ip-address* | *ipv6-address* | *peer-group-name*} **next-hop-unchanged** [**allpaths**]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**router bgp** *autonomous-system-number* **Example:** `Device(config)# router bgp 100`	Configures the BGP routing process.
Step 4	**neighbor** {*ip-address* | *ipv6-address* | *peer-group-name*} **remote-as** *as-number* **Example:** `Device(config-router)# neighbor 192.168.2.1 remote-as 100`	Adds an entry to the multiprotocol BGP neighbor table for eBGP peering with RR2.

	Command or Action	Purpose
Step 5	**neighbor** {*ip-address* | *ipv6-address* | *peer-group-name*} **update-source** *interface-type interface-number* **Example:** `Device(config-router)# neighbor 192.168.2.1 update-source Loopback 0`	Enables the BGP session to use a source address on the specified interface.
Step 6	**neighbor** {*ip-address* | *ipv6-address*| *peer-group-name*} **ebgp-multihop** [*ttl*] **Example:** `Device(config-router)# neighbor 192.168.2.1 ebgp-multihop`	(Optional) Accepts and attempts BGP connections to external peers residing on networks that are not directly connected.
Step 7	**address-family vpnv6** [**unicast**] **Example:** `Device(config-router)# address-family vpnv6`	(Optional) Places the device in address family configuration mode for configuring routing sessions.
Step 8	**neighbor** {*ip-address* | *peer-group-name* | *ipv6-address*} **activate** **Example:** `Device(config-router-af)# neighbor 192.168.2.1 activate`	Enables the exchange of information for this address family with the specified BGP neighbor.
Step 9	**neighbor** {*ip-address* | *ipv6-address* | **peer-group-name**} **send-community** [**both** | **standard** | **extended**] **Example:** `Device(config-router-af)# neighbor 192.168.2.1 send-community extended`	Specifies that a communities attribute should be sent to the BGP neighbor.
Step 10	**neighbor** {*ip-address* | *ipv6-address* | *peer-group-name*} **next-hop-unchanged** [**allpaths**] **Example:** `Device(config-router-af)# neighbor 192.168.2.1 next-hop-unchanged allpaths`	Enables an eBGP multihop peer to propagate to the next hop unchanged for paths.

Configuring the ASBR

- Configuring Peering with Router Reflector RR1, page 320

- Configuring Peering with the Other ISP ASBR2, page 321

Configuring Peering with Router Reflector RR1

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **router bgp** *autonomous-system-number*
4. **neighbor** {*ip-address* | *ipv6-address* | *peer-group-name*} **remote-as** *as-number*
5. **neighbor** {*ip-address* | *ipv6-address* | *peer-group-name*} **update-source** *interface-type interface-number*
6. **address-family ipv4** [**mdt** | **multicast** | **tunnel** | **unicast** [**vrf** *vrf-name*] | **vrf** *vrf-name*
7. **neighbor** {*ip-address* | *peer-group-name* | *ipv6-address*} **activate**
8. **neighbor** *ip-address* | *ipv6-address* | *peer-group-name* **send-label**
9. exit

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**router bgp** *autonomous-system-number* **Example:** `Device(config)# router bgp 100`	Configures the BGP routing process.
Step 4	**neighbor** {*ip-address* | *ipv6-address* | *peer-group-name*} **remote-as** *as-number* **Example:** `Device(config-router)# neighbor 192.168.2.115 remote-as 100`	Adds an entry to the multiprotocol BGP neighbor table for peering with RR1.

Command or Action	Purpose
Step 5 **neighbor** {*ip-address* | *ipv6-address* | *peer-group-name*} **update-source** *interface-type interface-number* **Example:** `Device(config-router)# neighbor 192.168.2.115 update-source Loopback 0`	Enables the BGP session to use a source address on the specified interface.
Step 6 **address-family ipv4** [**mdt** | **multicast** | **tunnel** | **unicast** [**vrf** *vrf-name*] | **vrf** *vrf-name* **Example:** `Device(config-router)# address-family ipv4`	Enters address family configuration mode to configure a routing session using standard IPv4 address prefixes.
Step 7 **neighbor** {*ip-address* | *peer-group-name* | *ipv6-address*} **activate** **Example:** `Device(config-router-af)# neighbor 192.168.2.115 activate`	Enables the exchange of information for this address family with the specified BGP neighbor.
Step 8 **neighbor** *ip-address* | *ipv6-address* | *peer-group-name* **send-label** **Example:** `Device(config-router-af)# neighbor 192.168.2.115 send-label`	Enables label exchange for this address family to this neighbor in order to send to the local PE the remote PE IPv4 loopback with a label in order to set up an end-to-end LSP.
Step 9 exit **Example:** `Device(config-router-af)# exit`	Exits address family configuration mode.

Configuring Peering with the Other ISP ASBR2

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **router bgp** *autonomous-system-number*
4. **neighbor** {*ip-address* | *ipv6-address* | *peer-group-name*} **remote-as** *as-number*
5. **neighbor** {*ip-address* | *ipv6-address* | *peer-group-name*} **update-source** *interface-type interface-number*
6. **neighbor** {*ip-address* | *ipv6-address*| *peer-group-name*} **ebgp-multihop** [*ttl*]
7. **address-family ipv4** [**mdt** | **multicast** | **tunnel** | **unicast** [**vrf** *vrf-name*] | **vrf** *vrf-name*]
8. **neighbor** {*ip-address* | *peer-group-name* | *ipv6-address*} **activate**
9. **neighbor** *ip-address* | *ipv6-address* | *peer-group-name*} **send-label**
10. **network** {*network-number* [**mask** *network-mask*] | *nsap-prefix*} [**route-map** *map-tag*]
11. **network** {*network-number* [**mask** *network-mask*] | *nsap-prefix*} [**route-map** *map-tag*]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**router bgp** *autonomous-system-number* **Example:** `Device(config)# router bgp 100`	Configures the BGP routing process.
Step 4	**neighbor** {*ip-address* | *ipv6-address* | *peer-group-name*} **remote-as** *as-number* **Example:** `Device(config-router)# neighbor 192.168.3.1 remote-as 100`	Adds an entry to the multiprotocol BGP neighbor table for peering with the ASBR2.

	Command or Action	Purpose
Step 5	**neighbor** {*ip-address* | *ipv6-address* | *peer-group-name*} **update-source** *interface-type interface-number* **Example:** `Device(config-router)# neighbor 192.168.3.1 update-source Loopback 0`	Enables the BGP session to use a source address on the specified interface.
Step 6	**neighbor** {*ip-address* | *ipv6-address*| *peer-group-name*} **ebgp-multihop** [*ttl*] **Example:** `Device(config-router)# neighbor 192.168.3.1 ebgp-multihop`	Accepts and attempts BGP connections to external peers residing on networks that are not directly connected.
Step 7	**address-family ipv4** [**mdt** | **multicast** | **tunnel** | **unicast** [**vrf** *vrf-name*] | **vrf** *vrf-name*] **Example:** `Device(config-router)# address-family ipv4`	Enters address family configuration mode to configure a routing session using standard IPv4 address prefixes.
Step 8	**neighbor** {*ip-address* | *peer-group-name* | *ipv6-address*} **activate** **Example:** `Device(config-router-af)# neighbor 192.168.3.1 activate`	Enables the exchange of information for this address family with the specified BGP neighbor.
Step 9	**neighbor** *ip-address* | *ipv6-address* | *peer-group-name*} **send-label** **Example:** `Device(config-router-af)# neighbor 192.168.3.1 send-label`	Enables label exchange for this address family to this neighbor in order to receive the remote PE IPv4 loopback with a label in order to set up an end-to-end LSP.
Step 10	**network** {*network-number* [**mask** *network-mask*] | *nsap-prefix*} [**route-map** *map-tag*] **Example:** `Device(config-router-af)# network 192.168.2.27 mask 255.255.255.255`	Flags a network as local to this autonomous system and enters the network to the BGP table. This configuration is for the PE VPN loopback.

	Command or Action	Purpose
Step 11	**network** {*network-number* [**mask** *network-mask*] | *nsap-prefix*} [**route-map** *map-tag*] **Example:** `Device(config-router-af)# network 192.168.2.15 mask 255.255.255.255`	Flags a network as local to this autonomous system and enters the network to the BGP table. This configuration is for the RR1 loopback.

Configuring CSC for IPv6 VPN

Perform this task to configure CsC-PE1 peering configuration with CsC-CE1.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **hostname** *name*
4. **router bgp** *autonomous-system-number*
5. **address-family ipv6** [**vrf** *vrf-name*] [**unicast** | **multicast**]
6. **neighbor** {*ip-address* | *ipv6-address* | *peer-group-name*} **remote-as** *as-number*
7. **neighbor** {*ip-address* | *peer-group-name* | *ipv6-address*} **activate**
8. **neighbor** { *ip-address* | *ipv6-address* | *peer-group-name*} **send-label**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**hostname** *name* **Example:** `Device(config)# hostname CSC-PE1`	Specifies or modifies the host name for the network server.

	Command or Action	Purpose
Step 4	**router bgp** *autonomous-system-number* **Example:** `Device(config)# router bgp 100`	Configures the BGP routing process.
Step 5	**address-family ipv6** [**vrf** *vrf-name*] [**unicast** | **multicast**] **Example:** `Device(config-router)# address-family ipv6 vrf ISP2`	Enters address family configuration mode.
Step 6	**neighbor** {*ip-address* | *ipv6-address* | *peer-group-name*} **remote-as** *as-number* **Example:** `Device(config-router-af)# neighbor FE80::866C:99 GigabitEthernet0/0/0 remote-as 200`	Adds an entry to the multiprotocol BGP neighbor table.
Step 7	**neighbor** {*ip-address* | *peer-group-name* | *ipv6-address*} **activate** **Example:** `Device(config-router-af)# neighbor FE80::866C:99 GigabitEthernet0/0/0 activate`	Enables the exchange of information for this address family with the specified BGP neighbor.
Step 8	**neighbor** { *ip-address* | *ipv6-address* | *peer-group-name*} **send-label** **Example:** `Device(config-router-af)# neighbor FE80::866C:99 GigabitEthernet0/0/0 send-label`	Enables label exchange for this address family to this neighbor.

Configuring BGP IPv6 PIC Edge for IP MPLS

Because many service provider networks contain many VRFs, the BGP PIC feature allows you to configure BGP PIC feature for all VRFs at once. Performing this task in IPv6 address family configuration mode protects IPv6 VRFs.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **router bgp** *autonomous-system-number*
4. **address-family ipv6** [**vrf** *vrf-name*] [**unicast** | **multicast** | **vpnv6**]
5. **bgp additional-paths install**
6. **bgp recursion host**

DETAILED STEPS

Command or Action	Purpose
Step 1 **enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2 **configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3 **router bgp** *autonomous-system-number* **Example:** `Router(config)# router bgp 100`	Configures the BGP routing process.
Step 4 **address-family ipv6** [**vrf** *vrf-name*] [**unicast**	**multicast**	**vpnv6**] **Example:** `Router(config-router)# address-family ipv6 vrf_pic`	Specifies a VRF table named vrf_pic, and enters IPv6 address family configuration mode.
Step 5 **bgp additional-paths install** **Example:** `Router(config-router-af)# bgp additional-paths install`	Calculates a backup path and installs it into the RIB and Cisco Express Forwarding.

	Command or Action	Purpose
Step 6	**bgp recursion host** **Example:** `Router(config-router-af)# bgp recursion host`	Enables the recursive-via-host flag for IPv6 address families.

Verifying and Troubleshooting IPv6 VPN

When users troubleshoot IPv6, any function that works similarly to VPNv4 will likely work for IPv6, therefore minimizing the learning curve for new IPv6 users. Few of the tools and commands used to troubleshoot 6PE and 6VPE are specific to IPv6; rather, the troubleshooting methodology is the same for both IPv4 and IPv6, and the commands and tools often vary by only one keyword.

Verifying and Troubleshooting Routing

Deploying 6PE and 6VPE involves principally BGP. The same set of commands used for VPNv4 can be used (with different set of arguments) for IPv6, and similar outputs are obtained.

Example: BGP IPv6 Activity Summary

```
Device# show bgp ipv6 summary

For address family: IPv6 Unicast
BGP router identifier 192.168.2.126, local AS number 33751
BGP table version is 15, main routing table version 15
12 network entries using 1692 bytes of memory
22 path entries using 1672 bytes of memory
5/4 BGP path/bestpath attribute entries using 580 bytes of memory
14 BGP rrinfo entries using 336 bytes of memory
2 BGP AS-PATH entries using 48 bytes of memory
0 BGP route-map cache entries using 0 bytes of memory
0 BGP filter-list cache entries using 0 bytes of memory
BGP using 4328 total bytes of memory
Dampening enabled. 0 history paths, 0 dampened paths
BGP activity 13/1 prefixes, 23/1 paths, scan interval 60 secs
Neighbor        V    AS MsgRcvd MsgSent   TblVer  InQ OutQ Up/Down  State/PfxRcd
192.168.2.146     4 33751     991     983       15    0    0 16:26:21       10
192.168.2.147     4 33751     991     983       15    0    0 16:26:22       10
FE80::4F6B:44 GigabitEthernet1/0/0
                4 20331     982     987       15    0    0 14:55:52        1
```

Example: Dumping the BGP IPv6 Tables

Each table (for example, BGP IPv6, BGP IPv6 VPN) can be reviewed individually, as shown in the following example:

```
Device# show bgp ipv6 unicast
BGP table version is 15, local router ID is 192.168.2.126
Status codes: s suppressed, d damped, h history, * valid, > best, i - internal,
              r RIB-failure, S Stale
Origin codes: i - IGP, e - EGP, ? - incomplete
   Network          Next Hop            Metric    LocPrf Weight Path
* i2001:DB8:100::/48 ::FFFF:192.168.2.101     0    100      0 10000 ?
*>i                  ::FFFF:192.168.2.101     0    100      0 10000 ?
* i2001:DB8::1/128   ::FFFF:192.168.2.101     0    100      0  i
*>i                  ::FFFF:192.168.2.101     0    100      0  i
```

Example: Dumping the IPv6 Routing Tables

IPv6 routing tables identify each routing protocol contributor to routable entries, as shown in the following example:

```
Device# show ipv6 route
IPv6 Routing Table - default - 13 entries
Codes: C - Connected, L - Local, S - Static, U - Per-user Static route
       B - BGP, R - RIP, I1 - ISIS L1, I2 - ISIS L2
       IA - ISIS interarea, IS - ISIS summary
       O - OSPF Intra, OI - OSPF Inter, OE1 - OSPF ext 1, OE2 - OSPF ext 2
       ON1 - OSPF NSSA ext 1, ON2 - OSPF NSSA ext 2
B   2001:DB8:100::/48 [200/0]
     via 192.168.2.101 Default-IP-Routing-Table, indirectly connected
B   2001:DB8::1/128 [200/0]
     via 192.168.2.101 Default-IP-Routing-Table, c
LC  2001:DB8::26/128 [0/0]
     via Loopback0, receive
```

From an IPv6 routing perspective, entries reachable over the MPLS backbone are listed as being indirectly connected, because MPLS is providing a Layer 2 tunnel mechanism.

Verifying and Troubleshooting Forwarding

Forwarding anomalies should be detected and understood so that users can perform troubleshooting. Commands such as **ping ipv6** and **traceroute ipv6** are used to validate data-plane connectivity and detect traffic black-holing. Commands such as **traceroute mpls**and **show mpls forwarding** can pinpoint a damaged node, interface, and forwarding error correction (FEC). At the edge, troubleshooting forwarding failures for a particular IPv6 destination commonly leads to breaking down the recursive resolution into elementary pieces. This task requires combining analysis of IPv6 routing (iBGP or eBGP), IP routing (IS-IS or OSPF), label distribution (BGP, LDP, or RSVP), and adjacency resolution to find a resolution breakage.

The following examples describe how to verify IPv6 VPN and troubleshoot various IPv6 VPN forwarding situations:

- Example: PE-CE Connectivity, page 328
- Example: PE Imposition Path, page 329
- Example: PE Disposition Path, page 331
- Example: Label Switch Path, page 331
- Example: VRF Information, page 332

Example: PE-CE Connectivity

The **ipv6 ping** and **traceroute** commands are useful to check connectivity from a PE to a CE, whether locally attached or remote over the MPLS backbone.

When a device is locally attached, one can use the **ipv6 ping** command with the CE link-local address (used for eBGP peering), as shown in the following example:

```
Device# ping FE80::4F6B:44%
Loopback0
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to FE80::4F6B:44, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 28/33/48 ms
```

The **ipv6 ping** command also can be used to test remote PE or CE reachability, but only IPv6 global addresses can be used (link-local addresses are not advertised beyond the link):

```
Device# ping 2001:DB8:1120:1::44
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 2001:DB8:1120:1:44::1, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 28/33/48 ms
```

Note that the **ping ipv6** and **traceroute** command functions over MPLS require PEs and CEs to announce one IPv6 global prefix. Each 6PE device announces 2001:DB8::PE#/128, filtered at the autonomous system edge. Each IPv6 CE configures 2001:DB8:*prefix*:CE#/128 and announces it as part as its less-specific prefix (2001:DB8:*prefix*::/n).

Reachability of remote PEs and CEs can be tested by using the **traceroute** command. If you have configured all PEs with the **no mpls ip propagate-ttl forwarded** command, when the **traceroute** command is executed from a CE, its output will show only the IPv6 nodes:

```
Device# traceroute 2001:DB8::1
Type escape sequence to abort.
Tracing the route to 2001:DB8::1
  1 2001:DB8::26 [AS 33751] 32 msec 32 msec 20 msec
  2 2001:DB8::1 [AS 33751] [MPLS: Label 73 Exp 0] 20 msec 20 msec 20 msec
  3 2001:DB8::1 [AS 33751] 28 msec 20 msec 20 msec
```

After the P devices have been upgraded with images that support ICMPv6, the **traceroute** command executed on the PE device (Time to Live [TTL] is then propagated) will also show P devices' responses, as shown in the following example:

```
Device# traceroute 2001:DB8::1

Type escape sequence to abort.
Tracing the route to 2001:DB8::1
  1 ::FFFF:172.20.25.1 [MPLS: Labels 38/73 Exp 0] 40 msec 32 msec 32 msec
  2 ::FFFF:172.20.10.1 [MPLS: Labels 30/73 Exp 0] 60 msec 32 msec 32 msec
  3 2001:DB8::1 [MPLS: Label 73 Exp 0] 32 msec 32 msec 16 msec
```

When run from a 6VPE device, both the **ping ipv6** and **traceroute** commands accept a *vrf* argument, exactly as in the case of VPNv4.

Note that the **traceroute** command is useful for evaluating the path across the MPLS backbone, but not for troubleshooting data-plane failures. The P devices are IPv6 unaware (and are also VPNv4 unaware), so the ICMPv6 messages that they generate in response to the **traceroute** command are forwarded to the egress PE using the received label stack. The egress PE can route the ICMPv6 message to the source of the traceroute. When the MPLS path is broken, it is also broken from the ICMP message, which cannot reach the egress PE.

Example: PE Imposition Path

On Cisco devices, the most useful tool for troubleshooting the imposition path for IPv6 is the **show ipv6 cef** command.

Dumping IPv6 Forwarding Table

You can use the **show ipv6 cef** command to display the forwarding table with label stacks used for each destination prefix, as shown in the following example:

```
Device# show ipv6 cef

2001:DB8:100::/48
  nexthop 172.20.25.1 GigabitEtherent0/0/0 label 38 72
2001:DB8::1/128
  nexthop 172.20.25.1 GigabitEtherent0/0/0 label 38 73
2001:DB8::26/128
  attached to Loopback0, receive
```

Details of an IPv6 Entry in the Forwarding Table

You can use the **show ipv6 cef** command to display details for a specific entry and to analyze how the destination was resolved and the label stack computed, as shown in the following example:

```
Device# show ipv6 cef 2001:DB8:100::/48 internal
2001:DB8:100::/48, epoch 0, RIB[B], refcount 4
  sources: RIB
..
  recursive via 192.168.2.101[IPv4:Default] label 72, fib 0252B1F8, 1 terminal fib
    path 024F56A8, path list 024F0BA8, share 0/1, type attached nexthop
    ifnums: (none)
     path_list contains at least one resolved destination(s). HW IPv4 notified.
    nexthop 172.20.25.1 GigabitEtherent0/0/0 label 38, adjacency IP adj out of
GigabitEtherent0/0/0 0289BEF0
  output chain: label 72 label 38 TAG adj out of GigabitEtherent0/0/0 0289BD80
```

Details of a BGP Entry in the BGP Table

The detailed output in the previous example shows that each label composing the label stack has a different origin that can be tracked down individually. The BGP table has the bottom label, as shown in the following example:

```
Device# show bgp ipv6 unicast 2001:DB8:100::/48

BGP routing table entry for 2001:DB8:100::/48, version 2
Paths: (2 available, best #2, table default)
  Advertised to update-groups:
     1
  10000
    ::FFFF:192.168.2.101 (metric 30) from 192.168.2.147 (192.168.2.147)
      Origin incomplete, metric 0, localpref 100, valid, internal
      Originator: 192.168.2.101, Cluster list: 192.168.2.147,
      mpls labels in/out nolabel/72
  10000
    ::FFFF:192.168.2.101 (metric 30) from 192.168.2.146 (192.168.2.146)
      Origin incomplete, metric 0, localpref 100, valid, internal, best
      Originator: 192.168.2.101, Cluster list: 192.168.2.146,
      mpls labels in/out nolabel/72
```

LDP, as shown in this example, displays the other labels:

```
Device# show mpls ldp bindings 192.168.2.101 32
  lib entry: 192.168.2.101/32, rev 56
        local binding:  label: 40
        remote binding: lsr: 192.168.2.119:0, label: 38
Device# show mpls ldp bindings 172.20.25.0 24
  lib entry: 172.20.25.0/24, rev 2
        local binding: label: imp-null
        remote binding: lsr: 192.168.2.119:0, label: imp-null
```

Example: PE Disposition Path

Use the following examples to troubleshoot the disposition path.

Dumping the MPLS Forwarding Table

The following example illustrates MPLS forwarding table information for troubleshooting the disposition path.

```
Device# show mpls forwarding-table
Local  Outgoing      Prefix            Bytes Label   Outgoing   Next Hop
Label  Label or VC   or Tunnel Id      Switched      interface
16     Pop Label     192.168.2.114/32    0             GE0/0/0   point2point
17     26            192.168.2.146/32    0             GE0/0/0   point2point
..
72     No Label      2001:DB8:100::/48   63121         GE1/0/0   point2point
73     Aggregate     2001:DB8::1/128   24123
```

BGP Label Analysis

The following example illustrates the label used for switching, which has been announced by iBGP (6PE in this example) and can be checked:

```
Device# show bgp ipv6 2001:DB8:100::/48

BGP routing table entry for 2001:DB8:100::/48, version 2
Paths: (1 available, best #1, table default)
  Advertised to update-groups:
     2
  10000
    FE80::2710:2 (FE80::2710:2) from FE80::2710:2 GigabitEthernet1/0/0 (192.168.2.103)
      Origin incomplete, metric 0, localpref 100, valid, external, best,
```

Example: Label Switch Path

Because the 6PE and 6VPE LSP endpoints are IPv4 addresses, the IPv4 tools for troubleshooting LSPs are useful for detecting data-plane failures that would lead to IPv6 traffic black-holing.

Analyzing the Label Switch Path

The following example displays the LSP IPv4 end:

```
Device# show ipv6 route 2001:DB8::1/128
Routing entry for 2001:DB8::1/128
  Known via "bgp 33751", distance 200, metric 0, type internal
  Route count is 1/1, share count 0
  Routing paths:
    192.168.2.101%Default-IP-Routing-Table indirectly connected
      MPLS Required
      Last updated 02:42:12 ago
```

Traceroute LSP Example

The following example shows the traceroute LSP:

```
Device# traceroute mpls ipv4 192.168.2.101/32 verbose

Tracing MPLS Label Switched Path to 192.168.2.101/32, timeout is 2 seconds
Codes: '!' - success, 'Q' - request not transmitted,
       '.' - timeout, 'U' - unreachable,
       'R' - downstream router but not target,
       'M' - malformed request
```

```
Type escape sequence to abort.
  0 172.20.25.2 0.0.0.0 MRU 1500 [Labels: 38 Exp: 0]
R 1 172.20.25.1 0.0.0.0 MRU 1500 [Labels: 30 Exp: 0] 40 ms, ret code 6
R 2 172.20.10.1 0.0.0.0 MRU 1504 [Labels: implicit-null Exp: 0] 60 ms, ret code 6
! 3 172.20.40.1 48 ms
```

Example: VRF Information

The following entries show VRF information for 6VPE.

show ipv6 cef vrf

The following is sample output from a Cisco Express Forwarding FIB associated with a VRF named cisco1:

```
Device# show ipv6 cef vrf cisco1
 2001:8::/64
  attached to GigabitEthernet0/0/1
 2001:8::3/128
  receive
 2002:8::/64
  nexthop 10.1.1.2 GigabitEthernet0/1/0 label 22 19
 2010::/64
  nexthop 2001:8::1 GigabitEthernet0/0/1
 2012::/64
  attached to Loopback1
 2012::1/128
  receive
```

show ipv6 route vrf

The following is sample output regarding an IPv6 routing table associated with a VRF named cisco1:

```
Device# show ipv6 route vrf cisco1
IPv6 Routing Table cisco1 - 6 entries
Codes: C - Connected, L - Local, S - Static, R - RIP, B - BGP
       U - Per-user Static route
       I1 - ISIS L1, I2 - ISIS L2, IA - ISIS interarea
       O - OSPF intra, OI - OSPF inter, OE1 - OSPF ext 1, OE2 - OSPF ext 2
C   2001:8::/64 [0/0]
     via ::, GigabitEthernet0/0/1
L   2001:8::3/128 [0/0]
     via ::, GigabitEthernet0/0/1
B   2002:8::/64 [200/0]
     via ::FFFF:192.168.1.4,
B   2010::/64 [20/1]
     via 2001:8::1,
C   2012::/64 [0/0]
     via ::, Loopback1
L   2012::1/128 [0/0]
     via ::, Loopback1
```

Debugging Routing and Forwarding

For troubleshooting of routing and forwarding anomalies, enabling debugging commands can prove useful, although several debug messages can slow the router and harm the usability of such a tool. For this reason, use **debug** commands with caution. The **debug ipv6 cef**, **debug mpls packet**, and **debug ipv6 packet** commands are useful for troubleshooting the forwarding path; the **debug bgp ipv6** and **debug bgp vpnv6** commands are useful for troubleshooting the control plane.

Configuration Examples for Implementing IPv6 VPN over MPLS

Example: IPv6 VPN Configuration Using IPv4 Next Hop

The following example illustrates a 6VPE next hop:

```
interface Loopback0
 ip address 192.168.2.11 255.255.255.255
!
router bgp 100
 neighbor 192.168.2.10 remote-as 100
 neighbor 192.168.2.10 update-source Loopback0
!
 address-family vpnv6
 neighbor 192.168.2.10 activate
 neighbor 192.168.2.10 send-community extended
 exit-address-family
```

By default, the next hop advertised will be the IPv6 VPN address:

```
[0:0]::FFFF:192.168.2.10
```

Note that it is a 192-bit address in the format of [RD]::FFFF:IPv4-address.

When the BGP IPv6 VPN peers share a common subnet, the MP_REACH_NLRI attribute contains a link-local address next hop in addition to the global address next hop. This situation typically occurs in an interautonomous-system topology when ASBRs are facing each other. In that case, the link-local next hop is used locally, and the global next hop is readvertised by BGP.

The BGP next hop is the keystone for building the label stack. The inner label is obtained from the BGP NLRI, and the outer label is the label distribution protocol (LDP) label to reach the IPv4 address embedded into the BGP next hop.

Additional References

Related Documents

Related Topic	Document Title
IPv6 Multiprotocol BGP	Implementing Multiprotocol BGP for IPv6
IPv6 EIGRP	Implementing EIGRP for IPv6
IPv6 MPLS	Implementing IPv6 over MPLS
IPv6 static routes	Implementing Static Routes for IPv6
IPv6 commands	*Cisco IOS IPv6 Command Reference*

Related Topic	Document Title
BGP PIC edge for IP and MPLS-VPN	"BGP PIC Edge for IP and MPLS-VPN ," *IP Routing: BGP Configuration Guide*

Standards

Standard	Title
draft-bonica-internet-icmp	*ICMP Extensions for Multiprotocol Label Switching*
draft-ietf-idr-bgp-ext-communities-0x.txt	*Cooperative Route Filtering Capability for BGP-4*

MIBs

MIB	MIBs Link
None	To locate and download MIBs for selected platforms, Cisco software releases, and feature sets, use Cisco MIB Locator found at the following URL: http://www.cisco.com/go/mibs

RFCs

RFC	Title
RFC 1267	*A Border Gateway Protocol 3 (BGP-3)*
RFC 1772	*Application of the Border Gateway Protocol in the Internet*
RFC 1918	*Address Allocation for Private Internets*
RFC 2547	*BGP/MPLS*
RFC 2858	*Multiprotocol Extensions for BGP-4*
RFC 3107	*Carrying Label Information in BGP-4*
RFC 3392	*Capabilities Advertisement with BGP-4*
RFC 3513	*Internet Protocol Version 6 (IPv6) Addressing Architecture*
RFC 4007	*IPv6 Scoped Address Architecture*
RFC 4193	*Unique Local IPv6 Unicast Addresses*
RFC 4364	*BGP MPLS/IP Virtual Private Networks (VPNs)*

RFC	Title
RFC 4382	*MPLS/BGP Layer 3 Virtual Private Network (VPN) Management Information Base*
RFC 4659	*BGP-MPLS IP Virtual Private Network (VPN) Extension for IPv6 VPN*

Technical Assistance

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

Feature Information for Implementing IPv6 VPN over MPLS

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Table 16 Feature Information for Implementing IPv6 VPN over MPLS

Feature Name	Releases	Feature Information
BGP IPv6 PIC Edge and Core for IP/MPLS	Cisco IOS XE Release 3.3S	The BGP IPv6 PIC Edge for IP/MPLS feature improves convergence after a network failure. The following commands were modified in this feature: **bgp additional-paths install**, **bgp advertise-best-external**, **bgp recursion host**.
IPv6 VPN over MPLS (6VPE)	Cisco IOS XE Release 3.1S	The IPv6 VPN (6VPE) over a MPLS IPv4 core infrastructure feature allows ISPs to offer IPv6 VPN services to their customers.

Feature Name	Releases	Feature Information
IPv6 VPN over MPLS (6VPE) InterAS Options	Cisco IOS XE Release 3.1S	This feature is supported in Cisco IOS XE Release 3.1.
MPLS VPN 6VPE Support over IP Tunnels	Cisco IOS XE Release 3.1S	This feature allows the use of IPv4 GRE tunnels to provide IPv6 VPN over MPLS functionality to reach the BGP next hop.

Glossary

- **6VPE device** —Provider edge device providing BGP-MPLS IPv6 VPN service over an IPv4-based MPLS core. It is a IPv6 VPN PE, dual-stack device that implements 6PE concepts on the core-facing interfaces.
- **customer edge (CE) device** —A service provider device that connects to VPN customer sites.
- **Forwarding Information Base (FIB)** —Table containing the information necessary to forward IP datagrams. At a minimum, the FIB contains the interface identifier and next-hop information for each reachable destination network prefix.
- **inbound route filtering (IRF)** —A BGP capability used for filtering incoming BGP updates that are not to be imported by the receiving PE device.
- **IPv6 provider edge device (6PE device)** —Device running a BGP-based mechanism to interconnect IPv6 islands over an MPLS-enabled IPv4 cloud.
- **IPv6 VPN address** —A IPv6 VPN address is a 24-byte identifier, beginning with an 8-byte route distinguisher (RD) and ending with a 16-byte IPv6 address. Sometimes it is called an IPv6 VPN address.
- **IPv6 VPN address family** —The address-family identifier (AFI) identifies a particular network-layer protocol and the subsequent AFI (SAFI) provides additional information. The AFI IPv6 SAFI VPN (AFI=2, SAFI=128) is called the IPv6 VPN address family. Sometimes it is called the IPv6 VPN address family. Similarly AFI IPv4 SAFI VPN is the VPNv4 address family.
- **network layer reachability information (NLRI)** —BGP sends routing update messages containing NLRI to describe a route and how to get there. In this context, an NLRI is a prefix. A BGP update message carries one or more NLRI prefixes and the attributes of a route for the NLRI prefixes; the route attributes include a BGP next hop gateway address and community values.
- **outbound route filtering (ORF)** —A BGP capability used to filtering outgoing BGP routing updates.
- **point of presence (POP)** —Physical location where an interexchange carrier installed equipment to interconnect with a local exchange carrier.
- **provider edge (PE) device** —A service provider device connected to VPN customer sites.
- **route distinguisher (RD)** —A 64-bit value prepended to an IPv6 prefix to create a globally unique IPv6 VPN address.
- **Routing Information Base (RIB)** —Also called the routing table.
- **Virtual routing and forwarding (VRF)** —A VPN routing and forwarding instance in a PE.
- **VRF table** —A routing and a forwarding table associated to a VRF. This is a customer-specific table that enables the PE device to maintain independent routing states for each customer.

Implementing IPv6 Multicast

Traditional IP communication allows a host to send packets to a single host (unicast transmission) or to all hosts (broadcast transmission). IPv6 multicast provides a third scheme, allowing a host to send a single data stream to a subset of all hosts (group transmission) simultaneously.

Finding Feature Information

Your software release may not support all the features documented in this module. For the latest caveats and feature information, see Bug Search Tool and the release notes for your platform and software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the feature information table at the end of this module.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Prerequisites for Implementing IPv6 Multicast

- In order to enable IPv6 multicast routing on a router, you must first enable IPv6 unicast routing on the router. For information on how to enable IPv6 unicast routing on a router, refer to Implementing IPv6 Addressing and Basic Connectivity.
- You must enable IPv6 unicast routing on all interfaces.
- This module assumes that you are familiar with IPv6 addressing and basic configuration. Refer to the Implementing IPv6 Addressing and Basic Connectivity module for more information.

Restrictions for Implementing IPv6 Multicast

- IPv6 multicast for Cisco IOS XE software uses Multicast Listener Discovery (MLD) version 2. This version of MLD is fully backward-compatible with MLD version 1 (described in RFC 2710). Hosts

that support only MLD version 1 will interoperate with a router running MLD version 2. Mixed LANs with both MLD version 1 and MLD version 2 hosts are likewise supported.

- When using bidirectional (bidir) range in a network, all routers in that network must be able to understand the bidirectional range in the bootstrap message (BSM).
- IPv6 multicast routing is disabled by default when the **ipv6 unicast-routing** command is configured.

Information About Implementing IPv6 Multicast

IPv6 Multicast Overview

An IPv6 multicast group is an arbitrary group of receivers that want to receive a particular data stream. This group has no physical or geographical boundaries--receivers can be located anywhere on the Internet or in any private network. Receivers that are interested in receiving data flowing to a particular group must join the group by signaling their local router. This signaling is achieved with the MLD protocol.

Routers use the MLD protocol to learn whether members of a group are present on their directly attached subnets. Hosts join multicast groups by sending MLD report messages. The network then delivers data to a potentially unlimited number of receivers, using only one copy of the multicast data on each subnet. IPv6 hosts that wish to receive the traffic are known as group members.

Packets delivered to group members are identified by a single multicast group address. Multicast packets are delivered to a group using best-effort reliability, just like IPv6 unicast packets.

The multicast environment consists of senders and receivers. Any host, regardless of whether it is a member of a group, can send to a group. However, only the members of a group receive the message.

A multicast address is chosen for the receivers in a multicast group. Senders use that address as the destination address of a datagram to reach all members of the group.

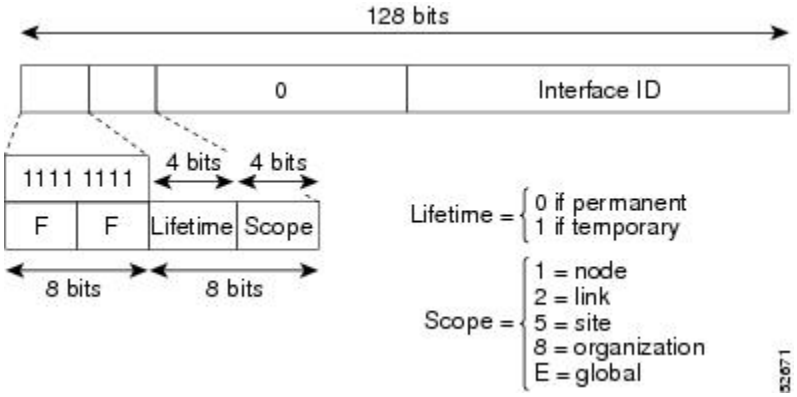
Membership in a multicast group is dynamic; hosts can join and leave at any time. There is no restriction on the location or number of members in a multicast group. A host can be a member of more than one multicast group at a time.

How active a multicast group is, its duration, and its membership can vary from group to group and from time to time. A group that has members may have no activity.

IPv6 Multicast Addressing

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. The figure below shows the format of the IPv6 multicast address.

Figure 34 IPv6 Multicast Address Format



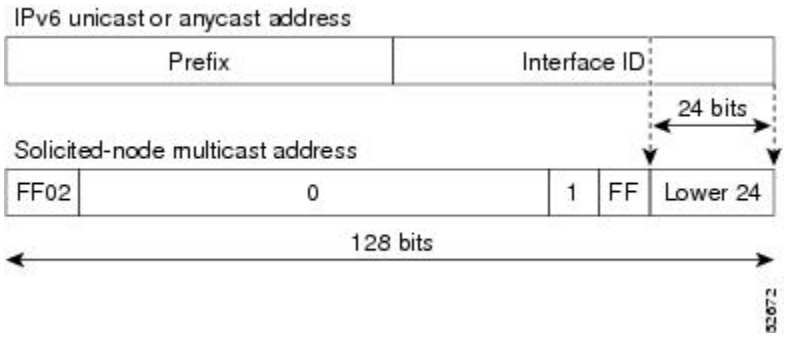
IPv6 nodes (hosts and routers) are required to join (receive packets destined for) the following multicast groups:

- 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 or anycast address (see the figure below). 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 35 IPv6 Solicited-Node Multicast Address Format



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

- IPv6 Multicast Groups, page 342

IPv6 Multicast Groups

An IPv6 address must be configured on an interface before the interface can forward IPv6 traffic. Configuring a site-local or 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 and anycast address assigned to the interface

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

- All-nodes link-local multicast group FF02::1
- All-routers link-local multicast group FF02::2

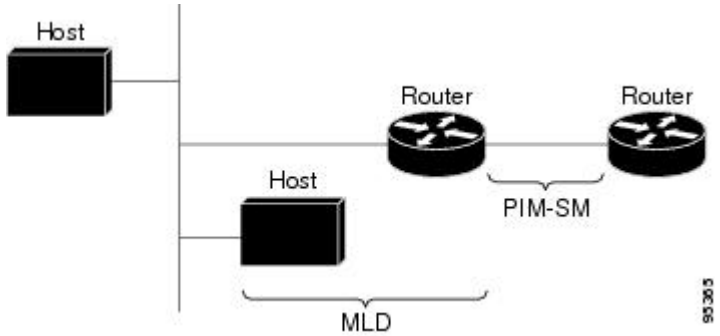
IPv6 Multicast Routing Implementation

Cisco software supports the following protocols to implement IPv6 multicast routing:

- MLD 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. There are two versions of MLD: MLD version 1 is based on version 2 of the Internet Group Management Protocol (IGMP) for IPv4, and MLD version 2 is based on version 3 of the IGMP for IPv4. IPv6 multicast for Cisco software uses both MLD version 2 and MLD version 1. MLD version 2 is fully backward-compatible with MLD version 1 (described in RFC 2710). Hosts that support only MLD version 1 will interoperate with a router running MLD version 2. Mixed LANs with both MLD version 1 and MLD version 2 hosts are likewise supported.
- PIM-SM is used between routers so that they can track which multicast packets to forward to each other and to their directly connected LANs.
- PIM in Source Specific Multicast (PIM-SSM) is similar to PIM-SM with the additional ability to report interest in receiving packets from specific source addresses (or from all but the specific source addresses) to an IP multicast address.

The figure below shows where MLD and PIM-SM operate within the IPv6 multicast environment.

Figure 36 IPv6 Multicast Routing Protocols Supported for IPv6



Multicast Listener Discovery Protocol for IPv6

To start implementing multicasting in the campus network, users must first define who receives the multicast. The MLD protocol is used by IPv6 routers to discover the presence of multicast listeners (for example, nodes that want to receive multicast packets) on their directly attached links, and to discover specifically which multicast addresses are of interest to those neighboring nodes. It is used for discovering local group and source-specific group membership. The MLD protocol provides a means to automatically control and limit the flow of multicast traffic throughout your network with the use of special multicast queriers and hosts.

The difference between multicast queriers and hosts is as follows:

- A querier is a network device, such as a router, that sends query messages to discover which network devices are members of a given multicast group.
- A host is a receiver, including routers, that send report messages to inform the querier of a host membership.

A set of queriers and hosts that receive multicast data streams from the same source is called a multicast group. Queriers and hosts use MLD reports to join and leave multicast groups and to begin receiving group traffic.

MLD uses the Internet Control Message Protocol (ICMP) to carry its messages. All MLD messages are link-local with a hop limit of 1, and they all have the router alert option set. The router alert option implies an implementation of the hop-by-hop option header.

MLD has three types of messages:

- Query--General, group-specific, and multicast-address-specific. In a query message, the multicast address field is set to 0 when MLD sends a general query. The general query learns which multicast addresses have listeners on an attached link.

Group-specific and multicast-address-specific queries are the same. A group address is a multicast address.

- Report--In a report message, the multicast address field is that of the specific IPv6 multicast address to which the sender is listening.
- Done--In a done message, the multicast address field is that of the specific IPv6 multicast address to which the source of the MLD message is no longer listening.

MLD states that result from MLD version 2 or MLD version 1 membership reports can be limited globally or by interface. The MLD group limits feature provides protection against denial of service (DoS) attacks caused by MLD packets. Membership reports in excess of the configured limits will not be entered in the MLD cache, and traffic for those excess membership reports will not be forwarded.

- MLD Access Group, page 343
- Explicit Tracking of Receivers, page 343

MLD Access Group

The MLD access group provides receiver access control in Cisco IOS XE IPv6 multicast routers. This feature limits the list of groups a receiver can join, and it allows or denies sources used to join SSM channels.

Explicit Tracking of Receivers

The explicit tracking feature allows a router to track the behavior of the hosts within its IPv6 network. This feature also enables the fast leave mechanism to be used with MLD version 2 host reports.

Protocol Independent Multicast

Protocol Independent Multicast (PIM) is used between routers so that they can track which multicast packets to forward to each other and to their directly connected LANs. PIM works independently of the unicast routing protocol to perform send or receive multicast route updates like other protocols. Regardless of which unicast routing protocols are being used in the LAN to populate the unicast routing table, Cisco IOS XE PIM uses the existing unicast table content to perform the Reverse Path Forwarding (RPF) check instead of building and maintaining its own separate routing table.

PIM-Sparse Mode

IPv6 multicast provides support for intradomain multicast routing using PIM-SM. PIM-SM uses unicast routing to provide reverse-path information for multicast tree building, but it is not dependent on any particular unicast routing protocol.

PIM-SM is used in a multicast network when relatively few routers are involved in each multicast and these routers do not forward multicast packets for a group, unless there is an explicit request for the traffic. PIM-SM distributes information about active sources by forwarding data packets on the shared tree. PIM-SM initially uses shared trees, which requires the use of an RP.

Requests are accomplished via PIM joins, which are sent hop by hop toward the root node of the tree. The root node of a tree in PIM-SM is the RP in the case of a shared tree or the first-hop router that is directly connected to the multicast source in the case of a shortest path tree (SPT). The RP keeps track of multicast groups and the hosts that send multicast packets are registered with the RP by that host’s first-hop router.

As a PIM join travels up the tree, routers along the path set up multicast forwarding state so that the requested multicast traffic will be forwarded back down the tree. When multicast traffic is no longer needed, a router sends a PIM prune up the tree toward the root node to prune (or remove) the unnecessary traffic. As this PIM prune travels hop by hop up the tree, each router updates its forwarding state appropriately. Ultimately, the forwarding state associated with a multicast group or source is removed.

A multicast data sender sends data destined for a multicast group. The designated router (DR) of the sender takes those data packets, unicast-encapsulates them, and sends them directly to the RP. The RP receives these encapsulated data packets, de-encapsulates them, and forwards them onto the shared tree. The packets then follow the (*, G) multicast tree state in the routers on the RP tree, being replicated wherever the RP tree branches, and eventually reaching all the receivers for that multicast group. The process of encapsulating data packets to the RP is called registering, and the encapsulation packets are called PIM register packets.

Designated Router

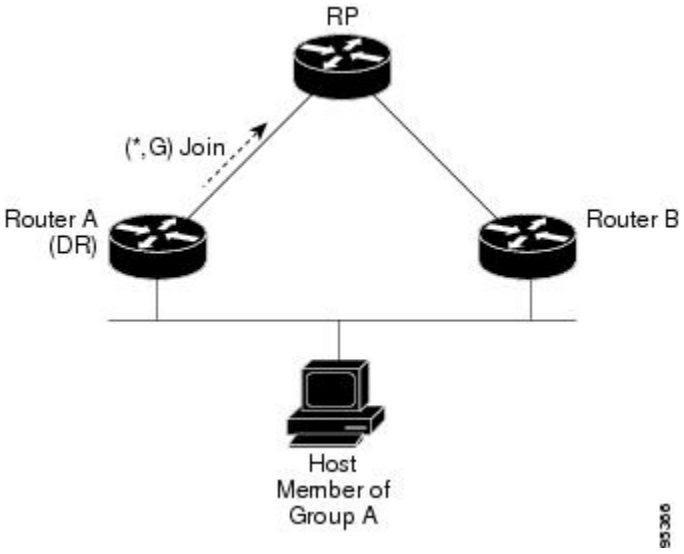
Cisco routers use PIM-SM to forward multicast traffic and follow an election process to select a designated router when more than one router is on a LAN segment.

The designated router is responsible for sending PIM register and PIM join and prune messages toward the RP to inform it about host group membership.

If multiple PIM-SM routers are on a LAN, a designated router must be elected to avoid duplicating multicast traffic for connected hosts. The PIM router with the highest IPv6 address becomes the DR for the LAN unless you choose to force the DR election by use of the **ipv6 pim dr-priority** command. This command allows you to specify the DR priority of each router on the LAN segment (default priority = 1) so that the router with the highest priority will be elected as the DR. If all routers on the LAN segment have the same priority, then the highest IPv6 address is again used as the tiebreaker.

The figure below illustrates what happens on a multiaccess segment. Router A and Router B are connected to a common multiaccess Gigabit Ethernet segment with Host A as an active receiver for Group A. Only Router A, operating as the DR, sends joins to the RP to construct the shared tree for Group A. If Router B was also permitted to send (*, G) joins to the RP, parallel paths would be created and Host A would receive duplicate multicast traffic. Once Host A begins to source multicast traffic to the group, the DR's responsibility is to send register messages to the RP. If both routers were assigned the responsibility, the RP would receive duplicate multicast packets.

Figure 37 Designated Router Election on a Multiaccess Segment



If the DR should fail, the PIM-SM provides a way to detect the failure of Router A and elect a failover DR. If the DR (Router A) became inoperable, Router B would detect this situation when its neighbor adjacency with Router A timed out. Because Router B has been hearing MLD membership reports from Host A, it already has MLD state for Group A on this interface and would immediately send a join to the RP when it became the new DR. This step reestablishes traffic flow down a new branch of the shared tree via Router B. Additionally, if Host A were sourcing traffic, Router B would initiate a new register process immediately after receiving the next multicast packet from Host A. This action would trigger the RP to join the SPT to Host A via a new branch through Router B.

Tip Two PIM routers are neighbors if there is a direct connection between them. To display your PIM neighbors, use the **show ipv6 pim neighbor** command in privileged EXEC mode.

Note DR election process is required only on multiaccess LANs. The last-hop router directly connected to the host is the DR.

Rendezvous Point

IPv6 PIM provides embedded RP support. Embedded RP support allows the router to learn RP information using the multicast group destination address instead of the statically configured RP. For routers that are the RP, the router must be statically configured as the RP.

The router searches for embedded RP group addresses in MLD reports or PIM messages and data packets. On finding such an address, the router learns the RP for the group from the address itself. It then uses this learned RP for all protocol activity for the group. For routers that are the RP, the router is advertised as an embedded RP must be configured as the RP.

To select a static RP over an embedded RP, the specific embedded RP group range or mask must be configured in the access list of the static RP. When PIM is configured in sparse mode, you must also choose one or more routers to operate as an RP. An RP is a single common root placed at a chosen point of a shared distribution tree and is configured statically in each box.

PIM DRs forward data from directly connected multicast sources to the RP for distribution down the shared tree. Data is forwarded to the RP in one of two ways:

- Data is encapsulated in register packets and unicast directly to the RP by the first-hop router operating as the DR.
- If the RP has itself joined the source tree, it is multicast-forwarded per the RPF forwarding algorithm described in the PIM-Sparse Mode section.

The RP address is used by first-hop routers to send PIM register messages on behalf of a host sending a packet to the group. The RP address is also used by last-hop routers to send PIM join and prune messages to the RP to inform it about group membership. You must configure the RP address on all routers (including the RP router).

A PIM router can be an RP for more than one group. Only one RP address can be used at a time within a PIM domain for a certain group. The conditions specified by the access list determine for which groups the router is an RP.

IPv6 multicast supports the PIM accept register feature, which is the ability to perform PIM-SM register message filtering at the RP. The user can match an access list or compare the AS path for the registered source with the AS path specified in a route map.

IPv6 BSR

PIM routers in a domain must be able to map each multicast group to the correct RP address. The BSR protocol for PIM-SM provides a dynamic, adaptive mechanism to distribute group-to-RP mapping information rapidly throughout a domain. With the IPv6 BSR feature, if an RP becomes unreachable, it will be detected and the mapping tables will be modified so that the unreachable RP is no longer used, and the new tables will be rapidly distributed throughout the domain.

Every PIM-SM multicast group needs to be associated with the IP or IPv6 address of an RP. When a new multicast sender starts sending, its local DR will encapsulate these data packets in a PIM register message

and send them to the RP for that multicast group. When a new multicast receiver joins, its local DR will send a PIM join message to the RP for that multicast group. When any PIM router sends a (*, G) join message, the PIM router needs to know which is the next router toward the RP so that the router can direct its (*, G) join message toward it. Also, when a PIM router is forwarding data packets using (*, G) state, the PIM router needs to know which is the correct incoming interface for packets destined for G, because it needs to reject any packets that arrive on other interfaces.

A small set of routers from a domain are configured as candidate bootstrap routers (C-BSRs) and a single BSR is selected for that domain. A set of routers within a domain are also configured as candidate RPs (C-RPs); typically, these routers are the same routers that are configured as C-BSRs. Candidate RPs periodically unicast candidate-RP-advertisement (C-RP-Adv) messages to the BSR of that domain, advertising their willingness to be an RP. A C-RP-Adv message includes the address of the advertising C-RP, and an optional list of group addresses and mask length fields, indicating the group prefixes for which the candidacy is advertised. The BSR then includes a set of these C-RPs, along with their corresponding group prefixes, in bootstrap messages (BSMs) it periodically originates. BSMs are distributed hop-by-hop throughout the domain.

The IPv6 BSR ability to configure RP mapping allows IPv6 multicast routers to be statically configured to announce scope-to-RP mappings directly from the BSR instead of learning them from candidate-RP messages. Announcing RP mappings from the BSR is useful in several situations:

- When an RP address never changes because there is only a single RP or the group range uses an anycast RP, it may be less complex to configure the RP address announcement statically on the candidate BSRs.
- When an RP address is a virtual RP address (such as when bidirectional PIM is used), it cannot be learned by the BSR from a candidate-RP. Instead, the virtual RP address must be configured as an announced RP on the candidate BSRs.

Cisco IOS XE IPv6 routers provide support for the RPF flooding of BSR packets so that a Cisco IOS XE IPv6 router will not disrupt the flow of BSMs. The router will recognize and parse enough of the BSM to identify the BSR address. The router performs an RPF check for this BSR address and forwards the packet only if it is received on the RPF interface. The router also creates a BSR entry containing RPF information to use for future BSMs from the same BSR. When BSMs from a given BSR are no longer received, the BSR entry is timed out.

Bidirectional BSR support allows bidirectional RPs to be advertised in C-RP messages and bidirectional ranges in the BSM. All routers in a system must be able to use the bidirectional range in the BSM; otherwise, the bidirectional RP feature will not function.

BSR provides scoped zone support by distributing group-to-RP mappings in networks using administratively scoped multicast. The user can configure candidate BSRs and a set of candidate RPs for each administratively scoped region in the user’s domain.

For BSR to function correctly with administrative scoping, a BSR and at least one C-RP must be within every administratively scoped region. Administratively scoped zone boundaries must be configured at the zone border routers (ZBRs), because they need to filter PIM join messages that might inadvertently cross the border due to error conditions. In addition, at least one C-BSR within the administratively scoped zone must be configured to be a C-BSR for the administratively scoped zone’s address range.

A separate BSR election will then take place (using BSMs) for every administratively scoped range, plus one for the global range. Administratively scoped ranges are identified in the BSM because the group range is marked to indicate that this is an administrative scope range, not just a range that a particular set of RPs is configured to handle.

Unless the C-RP is configured with a scope, it discovers the existence of the administratively scoped zone and its group range through reception of a BSM from the scope zone’s elected BSR containing the scope zone’s group range. A C-RP stores each elected BSR's address and the administratively scoped range

contained in its BSM. It separately unicasts C-RP-Adv messages to the appropriate BSR for every administratively scoped range within which it is willing to serve as an RP.

All PIM routers within a PIM bootstrap domain where administratively scoped ranges are in use must be able to receive BSMs and store the winning BSR and RP set for all administratively scoped zones that apply.

PIM-Source Specific Multicast

PIM-SSM is the routing protocol that supports the implementation of SSM and is derived from PIM-SM. However, unlike PIM-SM where data from all multicast sources are sent when there is a PIM join, the SSM feature forwards datagram traffic to receivers from only those multicast sources that the receivers have explicitly joined, thus optimizing bandwidth utilization and denying unwanted Internet broadcast traffic. Further, instead of the use of RP and shared trees, SSM uses information found on source addresses for a multicast group. This information is provided by receivers through the source addresses relayed to the last-hop routers by MLD membership reports, resulting in shortest-path trees directly to the sources.

In SSM, delivery of datagrams is based on (S, G) channels. Traffic for one (S, G) channel consists of datagrams with an IPv6 unicast source address S and the multicast group address G as the IPv6 destination address. Systems will receive this traffic by becoming members of the (S, G) channel. Signaling is not required, but receivers must subscribe or unsubscribe to (S, G) channels to receive or not receive traffic from specific sources.

MLD version 2 is required for SSM to operate. MLD allows the host to provide source information. Before SSM will run with MLD, SSM must be supported in the Cisco IPv6 router, the host where the application is running, and the application itself.

- SSM Mapping for IPv6, page 348
- PIM Shared Tree and Source Tree (Shortest-Path Tree), page 348
- Reverse Path Forwarding, page 350

SSM Mapping for IPv6

SSM mapping for IPv6 supports both static and dynamic Domain Name System (DNS) mapping for MLD version 1 receivers. This feature allows deployment of IPv6 SSM with hosts that are incapable of providing MLD version 2 support in their TCP/IP host stack and their IP multicast receiving application.

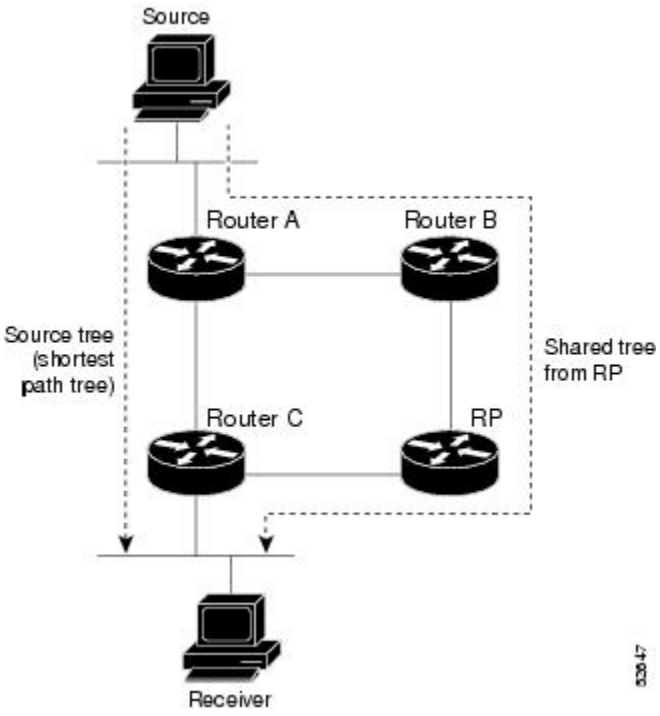
SSM mapping allows the router to look up the source of a multicast MLD version 1 report either in the running configuration of the router or from a DNS server. The router can then initiate an (S, G) join toward the source.

PIM Shared Tree and Source Tree (Shortest-Path Tree)

By default, members of a group receive data from senders to the group across a single data distribution tree rooted at the RP. This type of distribution tree is called shared tree or rendezvous point tree (RPT), as

illustrated in the figure below. Data from senders is delivered to the RP for distribution to group members joined to the shared tree.

Figure 38 Shared Tree and Source Tree (Shortest Path Tree)



If the data threshold warrants, leaf routers on the shared tree may initiate a switch to the data distribution tree rooted at the source. This type of distribution tree is called a shortest path tree or source tree. By default, the Cisco IOS XE software switches to a source tree upon receiving the first data packet from a source.

The following process details the move from shared tree to source tree:

1. Receiver joins a group; leaf Router C sends a join message toward the RP.
2. RP puts the link to Router C in its outgoing interface list.
3. Source sends the data; Router A encapsulates the data in the register and sends it to the RP.
4. RP forwards the data down the shared tree to Router C and sends a join message toward the source. At this point, data may arrive twice at Router C, once encapsulated and once natively.
5. When data arrives natively (unencapsulated) at the RP, the RP sends a register-stop message to Router A.
6. By default, receipt of the first data packet prompts Router C to send a join message toward the source.
7. When Router C receives data on (S, G), it sends a prune message for the source up the shared tree.
8. RP deletes the link to Router C from the outgoing interface of (S, G).
9. RP triggers a prune message toward the source.

Join and prune messages are sent for sources and RPs. They are sent hop-by-hop and are processed by each PIM router along the path to the source or RP. Register and register-stop messages are not sent hop-by-hop. They are sent by the designated router that is directly connected to a source and are received by the RP for the group.

Reverse Path Forwarding

Reverse-path forwarding is used for forwarding multicast datagrams. It functions as follows:

- If a router receives a datagram on an interface it uses to send unicast packets to the source, the packet has arrived on the RPF interface.
- If the packet arrives on the RPF interface, a router forwards the packet out the interfaces present in the outgoing interface list of a multicast routing table entry.
- If the packet does not arrive on the RPF interface, the packet is silently discarded to prevent loops.

PIM uses both source trees and RP-rooted shared trees to forward datagrams; the RPF check is performed differently for each, as follows:

- If a PIM router has source-tree state (that is, an (S, G) entry is present in the multicast routing table), the router performs the RPF check against the IPv6 address of the source of the multicast packet.
- If a PIM router has shared-tree state (and no explicit source-tree state), it performs the RPF check on the RP's address (which is known when members join the group).

Sparse-mode PIM uses the RPF lookup function to determine where it needs to send joins and prunes. (S, G) joins (which are source-tree states) are sent toward the source. (*, G) joins (which are shared-tree states) are sent toward the RP.

Routable Address Hello Option

When an IPv6 interior gateway protocol is used to build the unicast routing table, the procedure to detect the upstream router address assumes the address of a PIM neighbor is always same as the address of the next-hop router, as long as they refer to the same router. However, it may not be the case when a router has multiple addresses on a link.

Two typical situations can lead to this situation for IPv6. The first situation can occur when the unicast routing table is not built by an IPv6 interior gateway protocol such as multicast BGP. The second situation occurs when the address of an RP shares a subnet prefix with downstream routers (note that the RP router address has to be domain-wide and therefore cannot be a link-local address).

The routable address hello option allows the PIM protocol to avoid such situations by adding a PIM hello message option that includes all the addresses on the interface on which the PIM hello message is advertised. When a PIM router finds an upstream router for some address, the result of RPF calculation is compared with the addresses in this option, in addition to the PIM neighbor's address itself. Because this option includes all the possible addresses of a PIM router on that link, it always includes the RPF calculation result if it refers to the PIM router supporting this option.

Because of size restrictions on PIM messages and the requirement that a routable address hello option fits within a single PIM hello message, a limit of 16 addresses can be configured on the interface.

Bidirectional PIM

Bidirectional PIM allows multicast routers to keep reduced state information, as compared with unidirectional shared trees in PIM-SM. Bidirectional shared trees convey data from sources to the RPA and distribute them from the RPA to the receivers. Unlike PIM-SM, bidirectional PIM does not switch over to the source tree, and there is no register encapsulation of data from the source to the RP.

A single designated forwarder (DF) exists for each RPA on every link within a bidirectional PIM domain (including multiaccess and point-to-point links). The only exception is the RPL on which no DF exists. The DF is the router on the link with the best route to the RPA, which is determined by comparing MRIB-provided metrics. A DF for a given RPA forwards downstream traffic onto its link and forwards upstream

traffic from its link toward the rendezvous point link (RPL). The DF performs this function for all bidirectional groups that map to the RPA. The DF on a link is also responsible for processing Join messages from downstream routers on the link as well as ensuring that packets are forwarded to local receivers discovered through a local membership mechanism such as MLD.

Bidirectional PIM offers advantages when there are many moderate or low-rate sources. However, the bidirectional shared trees may have worse delay characteristics than do the source trees built in PIM-SM (depending on the topology).

Only static configuration of bidirectional RPs is supported in IPv6.

PIM Passive Mode

A router configured with PIM will always send out PIM hello messages to all interfaces enabled for IPv6 multicast routing, even if the router is configured not to accept PIM messages from any neighbor on the LAN. The IPv6 PIM passive mode feature allows PIM passive mode to be enabled on an interface so that a PIM passive interface cannot send and receive PIM control messages, but it can act as RPF interface for multicast route entries, and it can accept and forward multicast data packets.

Static Mroutes

IPv6 static mroutes behave much in the same way as IPv6 static routes. IPv6 static mroutes share the same database as IPv6 static routes and are implemented by extending static route support. Static mroutes support equal-cost multipath mroutes, and they also support unicast-only static routes.

MRIB

The Multicast Routing Information Base (MRIB) is a protocol-independent repository of multicast routing entries instantiated by multicast routing protocols (routing clients). Its main function is to provide independence between routing protocols and the Multicast Forwarding Information Base (MFIB). It also acts as a coordination and communication point among its clients.

Routing clients use the services provided by the MRIB to instantiate routing entries and retrieve changes made to routing entries by other clients. Besides routing clients, MRIB also has forwarding clients (MFIB instances) and special clients such as MLD. MFIB retrieves its forwarding entries from MRIB and notifies the MRIB of any events related to packet reception. These notifications can either be explicitly requested by routing clients or spontaneously generated by the MFIB.

Another important function of the MRIB is to allow for the coordination of multiple routing clients in establishing multicast connectivity within the same multicast session. MRIB also allows for the coordination between MLD and routing protocols.

MFIB

The MFIB is a platform-independent and routing-protocol-independent library for IPv6 software. Its main purpose is to provide a Cisco platform with an interface with which to read the IPv6 multicast forwarding table and notifications when the forwarding table changes. The information provided by the MFIB has clearly defined forwarding semantics and is designed to make it easy for the platform to translate to its specific hardware or software forwarding mechanisms.

When routing or topology changes occur in the network, the IPv6 routing table is updated, and those changes are reflected in the MFIB. The MFIB maintains next-hop address information based on the information in the IPv6 routing table. Because there is a one-to-one correlation between MFIB entries and

routing table entries, the MFIB contains all known routes and eliminates the need for route cache maintenance that is associated with switching paths such as fast switching and optimum switching.

- Distributed MFIB, page 352

Distributed MFIB

Distributed Multicast Forwarding Information Base (MFIB) is used to switch multicast IPv6 packets on distributed platforms. Distributed MFIB may also contain platform-specific information on replication across line cards. The basic MFIB routines that implement the core of the forwarding logic are common to all forwarding environments.

dMFIB implements the following functions:

- Distributes a copy of the MFIB to the line cards.
- Relays data-driven protocol events generated in the line cards to PIM.
- Provides an MFIB platform application program interface (API) to propagate MFIB changes to platform-specific code responsible for programming the hardware acceleration engine. This API also includes entry points to switch a packet in software (necessary if the packet is triggering a data-driven event) and to upload traffic statistics to the software.
- Provides hooks to allow clients residing on the RP to read traffic statistics on demand. Distributed MFIB does not periodically upload these statistics to the RP.

The combination of distributed MFIB and MRIB subsystems allows the device to have a "customized" copy of the MFIB database in each line card and to transport MFIB-related platform-specific information from the RP to the line cards.

IPv6 Multicast VRF Lite

The IPv6 Multicast VRF Lite feature provides IPv6 multicast support for multiple virtual routing/forwarding contexts (VRFs). The scope of these VRFs is limited to the router in which the VRFs are defined.

This feature provides separation between routing and forwarding, providing an additional level of security because no communication between devices belonging to different VRFs is allowed unless it is explicitly configured. The IPv6 Multicast VRF Lite feature simplifies the management and troubleshooting of traffic belonging to a specific VRF.

IPv6 Multicast Process Switching and Fast Switching

A unified MFIB is used to provide both fast switching and process switching support for PIM-SM and PIM-SSM in IPv6 multicast. In process switching, the Route Processor must examine, rewrite, and forward each packet. The packet is first received and copied into the system memory. The router then looks up the Layer 3 network address in the routing table. The Layer 2 frame is then rewritten with the next-hop destination address and sent to the outgoing interface. The RP also computes the cyclic redundancy check (CRC). This switching method is the least scalable method for switching IPv6 packets.

IPv6 multicast fast switching allows routers to provide better packet forwarding performance than process switching. Information conventionally stored in a route cache is stored in several data structures for IPv6 multicast switching. The data structures provide optimized lookup for efficient packet forwarding.

In IPv6 multicast forwarding, the first packet is fast-switched if the PIM protocol logic allows it. In IPv6 multicast fast switching, the MAC encapsulation header is precomputed. IPv6 multicast fast switching uses the MFIB to make IPv6 destination prefix-based switching decisions. In addition to the MFIB, IPv6

multicast fast switching uses adjacency tables to prepend Layer 2 addressing information. The adjacency table maintains Layer 2 next-hop addresses for all MFIB entries.

The adjacency table is populated as adjacencies are discovered. Each time an adjacency entry is created (such as through ARP), a link-layer header for that adjacent node is precomputed and stored in the adjacency table. Once a route is determined, it points to a next hop and corresponding adjacency entry. It is subsequently used for encapsulation during switching of packets.

A route might have several paths to a destination prefix, such as when a router is configured for simultaneous load balancing and redundancy. For each resolved path, a pointer is added for the adjacency corresponding to the next-hop interface for that path. This mechanism is used for load balancing across several paths.

Multiprotocol BGP for the IPv6 Multicast Address Family

The multiprotocol BGP for the IPv6 multicast address family feature provides multicast BGP extensions for IPv6 and supports the same features and functionality as IPv4 BGP. IPv6 enhancements to multicast BGP include support for an IPv6 multicast address family and network layer reachability information (NLRI) and next hop (the next router in the path to the destination) attributes that use IPv6 addresses.

Multicast BGP is an enhanced BGP that allows the deployment of interdomain IPv6 multicast. Multiprotocol BGP carries routing information for multiple network layer protocol address families; for example, IPv6 address family and for IPv6 multicast routes. The IPv6 multicast address family contains routes used for RPF lookup by the IPv6 PIM protocol, and multicast BGP IPv6 provides for interdomain transport of the same. Users must use multiprotocol BGP for IPv6 multicast when using IPv6 multicast with BGP because the unicast BGP learned routes will not be used for IPv6 multicast.

Multicast BGP functionality is provided through a separate address family context. A subsequent address family identifier (SAFI) provides information about the type of the network layer reachability information that is carried in the attribute. Multiprotocol BGP unicast uses SAFI 1 messages, and multiprotocol BGP multicast uses SAFI 2 messages. SAFI 1 messages indicate that the routes are usable only for IP unicast, not IP multicast. Because of this functionality, BGP routes in the IPv6 unicast RIB must be ignored in the IPv6 multicast RPF lookup.

A separate BGP routing table is maintained to configure incongruent policies and topologies (for example, IPv6 unicast and multicast) by using IPv6 multicast RPF lookup. Multicast RPF lookup is very similar to the IP unicast route lookup.

No MRIB is associated with the IPv6 multicast BGP table. However, IPv6 multicast BGP operates on the unicast IPv6 RIB when needed. Multicast BGP does not insert or update routes into the IPv6 unicast RIB.

Bandwidth-Based CAC for IPv6 Multicast

The bandwidth-based call admission control (CAC) for IPv6 multicast feature implements a way to count per-interface mroute state limiters using cost multipliers. This feature can be used to provide bandwidth-based CAC on a per-interface basis in network environments where the multicast flows use different amounts of bandwidth.

This feature limits and accounts for IPv6 multicast state in detail. When this feature is configured, interfaces can be limited to the number of times they may be used as incoming or outgoing interfaces in the IPv6 multicast PIM topology.

With this feature, router administrators can configure global limit cost commands for state matching access lists and specify which cost multiplier to use when accounting such state against the interface limits. This feature provides the required flexibility to implement bandwidth-based local CAC policy by tuning appropriate cost multipliers for different bandwidth requirements.

Threshold Notification for mCAC Limit

The threshold notification for mCAC limit feature notifies the user when actual simultaneous multicast channel numbers exceeds or fall below a specified threshold percentage. For example, if the mCAC rate limit is set to 50,000,000 and the configured threshold percentage is 80 percent, then the user is notified if the limit exceeds 10,000,000.

How to Implement IPv6 Multicast

Enabling IPv6 Multicast Routing

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ipv6 multicast-routing**

DETAILED STEPS

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

	Command or Action	Purpose
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**ipv6 multicast-routing** **Example:** `Router(config)# ipv6 multicast-routing`	Enables multicast routing on all IPv6-enabled interfaces and enables multicast forwarding for PIM and MLD on all enabled interfaces of the router.

Customizing and Verifying the MLD Protocol

Customizing and Verifying MLD on an Interface

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type number*
4. **ipv6 mld join-group** [*group-address*] [[**include** | **exclude**] {*source-address* | **source-list** [*acl*]}
5. **ipv6 mld access-group** *access-list-name*
6. **ipv6 mld static-group** [*group-address*] [[**include**| **exclude**] {*source-address* | **source-list** [*acl*]}
7. **ipv6 mld query-max-response-time** *seconds*
8. **ipv6 mld query-timeout** *seconds*
9. **ipv6 mld query-interval** *seconds*
10. **exit**
11. **show ipv6 mld groups** [**link-local**] [*group-name* | *group-address*] [*interface-type interface-number*] [**detail** | **explicit**
12. **show ipv6 mfib summary**
13. **show ipv6 mld interface** [*type number*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**interface** *type number* **Example:** `Router(config)# interface GigabitEthernet 1/0/0`	Specifies an interface type and number, and places the router in interface configuration mode.
Step 4	**ipv6 mld join-group** [*group-address*] [[**include** | **exclude**] {*source-address* | **source-list** [*acl*]} **Example:** `Router(config-if)# ipv6 mld join-group FF04::12 exclude 2001:DB8::10::11`	Configures MLD reporting for a specified group and source.
Step 5	**ipv6 mld access-group** *access-list-name* **Example:** `Router(config-if)# ipv6 access-list acc-grp-1`	Allows the user to perform IPv6 multicast receiver access control.
Step 6	**ipv6 mld static-group** [*group-address*] [[**include**| **exclude**] {*source-address* | **source-list** [*acl*]} **Example:** `Router(config-if)# ipv6 mld static-group ff04::10 include 100::1`	Statically forwards traffic for the multicast group onto a specified interface and cause the interface to behave as if a MLD joiner were present on the interface.

	Command or Action	Purpose
Step 7	**ipv6 mld query-max-response-time** *seconds* **Example:** `Router(config-if)# ipv6 mld query-max-response-time 20`	Configures the maximum response time advertised in MLD queries.
Step 8	**ipv6 mld query-timeout** *seconds* **Example:** `Router(config-if)# ipv6 mld query-timeout 130`	Configures the timeout value before the router takes over as the querier for the interface.
Step 9	**ipv6 mld query-interval** *seconds* **Example:** `Router(config-if)# ipv6 mld query-interval 60`	Configures the frequency at which the Cisco IOS XE software sends MLD host-query messages. **Caution** Changing this value may severely impact multicast forwarding.
Step 10	**exit** **Example:** `Router(config-if)# exit`	Enter this command twice to exit interface configuration mode and enter privileged EXEC mode.
Step 11	**show ipv6 mld groups** [**link-local**] [*group-name* | *group-address*] [*interface-type interface-number*] [**detail** | **explicit** **Example:** `Router# show ipv6 mld groups GigabitEthernet 2/1/0`	Displays the multicast groups that are directly connected to the router and that were learned through MLD.
Step 12	**show ipv6 mfib summary** **Example:** `Router# show ipv6 mfib summary`	Displays summary information about the number of IPv6 Multicast Forwarding Information Base (MFIB) entries (including link-local groups) and interfaces.
Step 13	**show ipv6 mld interface** [*type number* **Example:** `Router# show ipv6 mld interface GigabitEthernet 2/1/0`	Displays multicast-related information about an interface.

Implementing MLD Group Limits

Per-interface and global MLD limits operate independently of each other. Both per-interface and global MLD limits can be configured on the same router. The number of MLD limits, globally or per interface, is not configured by default; the limits must be configured by the user. A membership report that exceeds either the per-interface or the global state limit is ignored.

Implementing MLD Group Limits Globally

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ipv6 mld** [**vrf** *vrf-name*] **state-limit** *number*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**ipv6 mld** [**vrf** *vrf-name*] **state-limit** *number* **Example:** `Router(config)# ipv6 mld state-limit 300`	Limits the number of MLD states globally.

Implementing MLD Group Limits per Interface

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type number*
4. **ipv6 mld limit** *number* [**except** *access-list*]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**interface** *type number* **Example:** `Router(config)# interface GigabitEthernet 1/0/0`	Specifies an interface type and number, and places the router in interface configuration mode.
Step 4	**ipv6 mld limit** *number* [**except** *access-list*] **Example:** `Router(config-if)# ipv6 mld limit 100`	Limits the number of MLD states on a per-interface basis.

Configuring Explicit Tracking of Receivers to Track Host Behavior

The explicit tracking feature allows a router to track the behavior of the hosts within its IPv6 network and enables the fast leave mechanism to be used with MLD version 2 host reports.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type number*
4. **ipv6 mld explicit-tracking** *access-list-name*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**interface** *type number* **Example:** `Router(config)# interface GigabitEthernet 1/0/0`	Specifies an interface type and number, and places the router in interface configuration mode.
Step 4	**ipv6 mld explicit-tracking** *access-list-name* **Example:** `Router(config-if)# ipv6 mld explicit-tracking list1`	Enables explicit tracking of hosts.

Disabling the Router from Receiving Unauthenticated Multicast Traffic

In some situations, access control may be needed to prevent multicast traffic from being received unless the subscriber is authenticated and the channels are authorized as per access control profiles. That is, there should be no traffic at all unless specified otherwise by access control profiles.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ipv6 multicast group-range** [*access-list-name*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**ipv6 multicast group-range** [*access-list-name* **Example:** `Router(config)# ipv6 multicast group-range`	Disables multicast protocol actions and traffic forwarding for unauthorized groups or channels on all the interfaces in a router.

Resetting the MLD Traffic Counters

SUMMARY STEPS

1. **enable**
2. **clear ipv6 mld** [**vrf** *vrf-name*] traffic
3. **show ipv6 mld** [**vrf** *vrf-name*] traffic

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**clear ipv6 mld** [**vrf** *vrf-name*] traffic **Example:** `Router# clear ipv6 mld traffic`	Resets all MLD traffic counters.

	Command or Action	Purpose
Step 3	**show ipv6 mld** [**vrf** *vrf-name*] traffic **Example:** `Router# show ipv6 mld traffic`	Displays the MLD traffic counters.

Clearing the MLD Interface Counters

SUMMARY STEPS

1. **enable**
2. **clear ipv6 mld** [**vrf** *vrf-name*] **counters** *interface-type*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**clear ipv6 mld** [**vrf** *vrf-name*] **counters** *interface-type* **Example:** `Router# clear ipv6 mld counters GigabitEthernet1/0/0`	Clears the MLD interface counters.

Configuring PIM

Configuring PIM Options

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ipv6 pim** [**vrf** *vrf-name*] **spt-threshold infinity** [**group-list** *access-list-name*]
4. **interface** *type number*
5. **ipv6 pim dr-priority** *value*
6. **ipv6 pim hello-interval** *seconds*
7. **ipv6 pim join-prune-interval** *seconds*
8. **exit**
9. **show ipv6 pim** [**vrf** *vrf-name*] **join-prune statistic** [*interface-type*]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**ipv6 pim** [**vrf** *vrf-name*] **spt-threshold infinity** [**group-list** *access-list-name*] **Example:** `Router(config)# ipv6 pim spt-threshold infinity group-list acc-grp-1`	Configures when a PIM leaf router joins the SPT for the specified groups.
Step 4	**interface** *type number* **Example:** `Router(config)# interface GigabitEthernet 1/0/0`	Specifies an interface type and number, and places the router in interface configuration mode.
Step 5	**ipv6 pim dr-priority** *value* **Example:** `Router(config-if)# ipv6 pim dr-priority 3`	Configures the DR priority on a PIM router.

	Command or Action	Purpose
Step 6	**ipv6 pim hello-interval** *seconds* **Example:** `Router(config-if)# ipv6 pim hello-interval 45`	Configures the frequency of PIM hello messages on an interface.
Step 7	**ipv6 pim join-prune-interval** *seconds* **Example:** `Router(config-if)# ipv6 pim join-prune-interval 75`	Configures periodic join and prune announcement intervals for a specified interface.
Step 8	**exit** **Example:** `Router(config-if)# exit`	Enter this command twice to exit interface configuration mode and enter privileged EXEC mode.
Step 9	**show ipv6 pim** [**vrf** *vrf-name*] **join-prune statistic** [*interface-type*] **Example:** `Router# show ipv6 pim join-prune statistic`	Displays the average join-prune aggregation for the most recently aggregated 1000, 10,000, and 50,000 packets for each interface.

Configuring Bidirectional PIM and Displaying Bidirectional PIM Information

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ipv6 pim** [**vrf** *vrf-name*] **rp-address** *ipv6-address* [*group-access-list*] [**bidir**]
4. **exit**
5. **show ipv6 pim df** [*interface-type interface-number*] [*rp-address*]
6. **show ipv6 pim** [**vrf** *vrf-name*] **df winner**[*interface-type interface-number*] [*rp-address*]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**ipv6 pim** [**vrf** *vrf-name*] **rp-address** *ipv6-address* [*group-access-list*] [**bidir**] **Example:** `Router(config)# ipv6 pim rp-address 2001:DB8::01:800:200E:8C6C bidir`	Configures the address of a PIM RP for a particular group range. Use of the **bidir** keyword means that the group range will be used for bidirectional shared-tree forwarding.
Step 4	**exit** **Example:** `Router(config-if)# exit`	Exits global configuration mode, and returns the router to privileged EXEC mode.
Step 5	**show ipv6 pim df** [*interface-type interface-number*] [*rp-address*] **Example:** `Router# show ipv6 pim df`	Displays the designated forwarder (DF)-election state of each interface for RP.
Step 6	**show ipv6 pim** [**vrf** *vrf-name*] **df winner**[*interface-type interface-number*] [*rp-address*] **Example:** `Router# show ipv6 pim df winner GigabitEthernet 1/0/0 200::1`	Displays the DF-election winner on each interface for each RP.

Configuring IPv6 PIM Passive Mode

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ipv6 multicast pim-passive-enable**
4. **interface** *type number*
5. **ipv6 pim passive**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**ipv6 multicast pim-passive-enable** **Example:** `Router(config)# ipv6 multicast pim-passive-enable`	Enables the PIM passive feature on an IPv6 router.
Step 4	**interface** *type number* **Example:** `Router(config)# interface GigabitEthernet 1/0/0`	Specifies an interface type and number, and places the router in interface configuration mode.
Step 5	**ipv6 pim passive** **Example:** `Router(config-if)# ipv6 pim passive`	Enables the PIM passive feature on a specific interface.

Resetting the PIM Traffic Counters

If PIM malfunctions or in order to verify that the expected number of PIM packets are received and sent, the user can clear PIM traffic counters. Once the traffic counters are cleared, the user can enter the **show ipv6 pim traffic** command to verify that PIM is functioning correctly and that PIM packets are being received and sent correctly.

SUMMARY STEPS

1. **enable**
2. **clear ipv6 pim** [**vrf** *vrf-name*] **traffic**
3. **show ipv6 pim** [**vrf** *vrf-name*] **traffic**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**clear ipv6 pim** [**vrf** *vrf-name*] **traffic** **Example:** `Router# clear ipv6 pim traffic`	Resets the PIM traffic counters.
Step 3	**show ipv6 pim** [**vrf** *vrf-name*] **traffic** **Example:** `Router# show ipv6 pim traffic`	Displays the PIM traffic counters.

Clearing the PIM Topology Table to Reset the MRIB Connection

No configuration is necessary to use the MRIB. However, users may in certain situations want to clear the PIM topology table in order to reset the MRIB connection, and verify MRIB information.

SUMMARY STEPS

1. **enable**
2. **clear ipv6 pim topology** [*group-name* | *group-address*
3. **show ipv6 mrib client filter**] [**name** {*client-name* | *client-name* **:** *client-id*}]
4. **show ipv6 mrib route** [**link-local** | **summary** | *source-address* | *source-name* | *****] [*group-name* | *group-address* [*prefix-length*]]
5. **show ipv6 pim topology** [**link-local** | **route-count** | *group-name* | *group-address*] [*source-address* | *source-name*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**clear ipv6 pim topology** [*group-name* | *group-address* **Example:** `Router# clear ipv6 pim topology FF04::10`	Clears the PIM topology table.
Step 3	**show ipv6 mrib client filter**] [**name** {*client-name* | *client-name* **:** *client-id*}] **Example:** `Router# show ipv6 mrib client`	Displays multicast-related information about an interface.
Step 4	**show ipv6 mrib route** [**link-local** | **summary** | *source-address* | *source-name* | ***] [*group-name* | *group-address* [*prefix-length*]] **Example:** `Router# show ipv6 mrib route`	Displays the MRIB route information.
Step 5	**show ipv6 pim topology** [**link-local** | **route-count** | *group-name* | *group-address*] [*source-address* | *source-name* **Example:** `Router# show ipv6 pim topology`	Displays PIM topology table information for a specific group or all groups.

Configuring a BSR

Configuring a BSR and Verifying BSR Information

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ipv6 pim** [**vrf** *vrf-name*] **bsr candidate bsr** *ipv6-address*[*hash-mask-length*] [**priority** *priority-value*]
4. **interface** *type number*
5. **ipv6 pim bsr border**
6. **exit**
7. **show ipv6 pim** [**vrf** *vrf-name*] **bsr** {**election** | **rp-cache** | **candidate-rp**}

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**ipv6 pim** [**vrf** *vrf-name*] **bsr candidate bsr** *ipv6-address*[*hash-mask-length*] [**priority** *priority-value*] **Example:** `Router(config)# ipv6 pim bsr candidate bsr 2001:DB8:3000:3000::42 124 priority 10`	Configures a router to be a candidate BSR.
Step 4	**interface** *type number* **Example:** `Router(config)# interface GigabitEthernet 1/0/0`	Specifies an interface type and number, and places the router in interface configuration mode.
Step 5	**ipv6 pim bsr border** **Example:** `Router(config-if)# ipv6 pim bsr border`	Configures a border for all BSMs of any scope on a specified interface.

	Command or Action	Purpose
Step 6	**exit** **Example:** `Router(config-if)# exit`	Enter this command twice to exit interface configuration mode and enter privileged EXEC mode.
Step 7	**show ipv6 pim** [**vrf** *vrf-name*] **bsr** {**election** | **rp-cache** | **candidate-rp**} **Example:** `Router# show ipv6 pim bsr election`	Displays information related to PIM BSR protocol processing.

Sending PIM RP Advertisements to the BSR

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ipv6 pim** [**vrf** *vrf-name*] **bsr candidate rp** *ipv6-address* [**group-list** *access-list-name*] [**priority** *priority-value*] [**interval** *seconds*] [**scope** *scope-value*] [**bidir**]
4. **interface** *type number*
5. **ipv6 pim bsr border**

DETAILED STEPS

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

Command or Action	Purpose
Step 3 **ipv6 pim** [**vrf** *vrf-name*] **bsr candidate rp** *ipv6-address* [**group-list** *access-list-name*] [**priority** *priority-value*] [**interval** *seconds*] [**scope** *scope-value*] [**bidir**] **Example:** `Router(config)# ipv6 pim bsr candidate rp 2001:DB8:3000:3000::42 priority 0`	Sends PIM RP advertisements to the BSR.
Step 4 **interface** *type number* **Example:** `Router(config)# interface GigabitEthernet 1/0/0`	Specifies an interface type and number, and places the router in interface configuration mode.
Step 5 **ipv6 pim bsr border** **Example:** `Router(config-if)# ipv6 pim bsr border`	Configures a border for all BSMs of any scope on a specified interface.

- Disabling the Router from Receiving Unauthenticated Multicast Traffic, page 371

Disabling the Router from Receiving Unauthenticated Multicast Traffic

In some situations, access control may be needed to prevent multicast traffic from being received unless the subscriber is authenticated and the channels are authorized as per access control profiles. That is, there should be no traffic at all unless specified otherwise by access control profiles.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ipv6 multicast group-range** [*access-list-name*]

DETAILED STEPS

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

	Command or Action	Purpose
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**ipv6 multicast group-range** [*access-list-name*] **Example:** `Router(config)# ipv6 multicast group-range`	Disables multicast protocol actions and traffic forwarding for unauthorized groups or channels on all the interfaces in a router.

Configuring SSM Mapping

When the SSM mapping feature is enabled, DNS-based SSM mapping is automatically enabled, which means that the router will look up the source of a multicast MLD version 1 report from a DNS server.

You can use either DNS-based or static SSM mapping, depending on your router configuration. If you choose to use static SSM mapping, you can configure multiple static SSM mappings. If multiple static SSM mappings are configured, the source addresses of all matching access lists will be used.

Note To use DNS-based SSM mapping, the router needs to find at least one correctly configured DNS server, to which the router may be directly attached.

>

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ipv6 mld** [**vrf** *vrf-name*] **ssm-map enable**
4. **no ipv6 mld** [**vrf** *vrf-name*] **ssm-map query dns**
5. **ipv6 mld** [**vrf** *vrf-name*] **ssm-map static** *access-list source-address*
6. **exit**
7. **show ipv6 mld** [**vrf** *vrf-name*] **ssm-map** [*source-address*]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**ipv6 mld** [**vrf** *vrf-name*] **ssm-map enable** **Example:** `Router(config)# ipv6 mld ssm-map enable`	Enables the SSM mapping feature for groups in the configured SSM range.
Step 4	**no ipv6 mld** [**vrf** *vrf-name*] **ssm-map query dns** **Example:** `Router(config)# no ipv6 mld ssm-map query dns`	Disables DNS-based SSM mapping.
Step 5	**ipv6 mld** [**vrf** *vrf-name*] **ssm-map static** *access-list source-address* **Example:** `Router(config)# ipv6 mld ssm-map static SSM_MAP_ACL_2 2001:DB8:1::1`	Configures static SSM mappings.
Step 6	**exit** **Example:** `Router(config-if)# exit`	Exits global configuration mode, and returns the router to privileged EXEC mode.
Step 7	**show ipv6 mld** [**vrf** *vrf-name*] **ssm-map** [*source-address*] **Example:** `Router# show ipv6 mld ssm-map`	Displays SSM mapping information.

Configuring Static Mroutes

Static multicast routes (mroutes) in IPv6 can be implemented as an extension of IPv6 static routes. You can configure your router to use a static route for unicast routing only, to use a static multicast route for multicast RPF selection only, or to use a static route for both unicast routing and multicast RPF selection.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ipv6 route** *ipv6-prefix / prefix-length ipv6-address | interface-type interface-number ipv6-address*]} [*administrative-distance*] [*administrative-multicast-distance* | **unicast**| **multicast**] [**tag** *tag*
4. **exit**
5. **show ipv6 mroute** [**vrf** *vrf-name*] [**link-local** | [*group-name* | *group-address* [*source-address* | *source-name*]] [**summary**] [**count**]
6. **show ipv6 mroute** [**vrf** *vrf-name*] [**link-local** | *group-name* | *group-address*] **active**[*kbps*]
7. **show ipv6 rpf** [**vrf** *vrf-name*] *ipv6-prefix*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**ipv6 route** *ipv6-prefix / prefix-length ipv6-address	interface-type interface-number ipv6-address*]} [*administrative-distance*] [*administrative-multicast-distance*	**unicast**| **multicast**] [**tag** *tag* **Example:** `Router(config)# ipv6 route 2001:DB8::/64 6::6 100`	Establishes static IPv6 routes. The example shows a static route used for both unicast routing and multicast RPF selection.
Step 4	**exit** **Example:** `Router(config-if)# exit`	Exits global configuration mode, and returns the router to privileged EXEC mode.

	Command or Action	Purpose
Step 5	**show ipv6 mroute** [**vrf** *vrf-name*] [**link-local** | [*group-name* | *group-address* [*source-address* | *source-name*]] [**summary**] [**count**] **Example:** `Router# show ipv6 mroute ff07::1`	Displays the contents of the IPv6 multicast routing table.
Step 6	**show ipv6 mroute** [**vrf** *vrf-name*] [**link-local** | *group-name* | *group-address*] **active**[*kbps*] **Example:** `Router# show ipv6 mroute active`	Displays the active multicast streams on the router.
Step 7	**show ipv6 rpf** [**vrf** *vrf-name*] *ipv6-prefix* **Example:** `Router# show ipv6 rpf 2001:DB8::1:1:2`	Checks RPF information for a given unicast host address and prefix.

Configuring IPv6 Multiprotocol BGP

Configuring an IPv6 Peer Group to Perform Multicast BGP Routing

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **router bgp** *as-number*
4. **neighbor** *peer-group-name* **peer-group**
5. **neighbor** {*ip-address* | *ipv6-address* | *peer-group-name*} **remote-as** *as-number*
6. **address-family ipv6** [**unicast** | **multicast**]
7. **neighbor** {*ip-address* | *peer-group-name* | *ipv6-address*} **activate**
8. **neighbor** {*ip-address* | *ipv6-address*} **peer-group** *peer-group-name*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**router bgp** *as-number* **Example:** `Device(config)# router bgp 65000`	Enters router configuration mode for the specified BGP routing process.
Step 4	**neighbor** *peer-group-name* **peer-group** **Example:** `Device(config-router)# neighbor group1 peer-group`	Creates a BGP peer group.
Step 5	**neighbor** {*ip-address* | *ipv6-address* | *peer-group-name*} **remote-as** *as-number* **Example:** `Device(config-router)# neighbor 2001:DB8:0:CC00::1 remote-as 64600`	Adds the IPv6 address of the neighbor in the specified autonomous system to the IPv6 multicast BGP neighbor table of the local router. • The *ipv6-address* argument in the **neighbor remote-as** command must be in the form documented in RFC 2373 where the address is specified in hexadecimal using 16-bit values between colons.

Command or Action	Purpose
Step 6 **address-family ipv6** [**unicast** | **multicast**] **Example:** `Device(config-router)# address-family ipv6 multicast`	Specifies the IPv6 address family, and enters address family configuration mode. • The **unicast** keyword specifies the IPv6 unicast address family. By default, the router is placed in configuration mode for the IPv6 unicast address family if a keyword is not specified in the **address-family ipv6** command. • The **multicast** keyword specifies IPv6 multicast address prefixes.
Step 7 **neighbor** {*ip-address* | *peer-group-name* | *ipv6-address*} **activate** **Example:** `Device(config-router-af)# neighbor 2001:DB8:0:CC00::1 activate`	Enables the neighbor to exchange prefixes for the specified family type with the neighbor and the local router. • To avoid extra configuration steps for each neighbor, use the **neighbor activate** command with the *peer-group-name* argument as an alternative in this step.
Step 8 **neighbor** {*ip-address* | *ipv6-address*} **peer-group** *peer-group-name* **Example:** `Device(config-router-af)# neighbor 2001:DB8:0:CC00::1 peer-group group1`	Assigns the IPv6 address of a BGP neighbor to a peer group.

• What to Do Next, page 377

What to Do Next

Refer to the section "Configuring an IPv6 Multiprotocol BGP Peer Group" in the Implementing Multiprotocol BGP for IPv6 module and the "Configure BGP Peer Groups" section of the "Configuring BGP" chapter in the *Cisco IOS XE IP Routing Configuration Guide* , for more information on assigning options to peer groups and making a BGP or multicast BGP neighbor a member of a peer group.

Advertising Routes into IPv6 Multiprotocol BGP

By default, networks that are defined in router configuration mode using the **network** command are injected into the IPv4 unicast database. To inject a network into another database, such as the IPv6 BGP database, you must define the network using the **network** command in address family configuration mode for the other database, as shown for the IPv6 BGP database.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **router bgp** *as-number*
4. **address-family ipv6** [**vrf** *vrf-name*] [**unicast** | **multicast** | **vpnv6**]
5. **network** {*network-number* [**mask** *network-mask*] | *nsap-prefix*} [**route-map** *map-tag*]
6. **exit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**router bgp** *as-number* **Example:** `Device(config)# router bgp 65000`	Enters router configuration mode for the specified BGP routing process.
Step 4	**address-family ipv6** [**vrf** *vrf-name*] [**unicast**	**multicast**	**vpnv6**] **Example:** `Device(config-router)# address-family ipv6 unicast`	Specifies the IPv6 address family, and enters address family configuration mode. • The **unicast** keyword specifies the IPv6 unicast address family. By default, the router is placed in configuration mode for the IPv6 unicast address family if a keyword is not specified with the **address-family ipv6** command. • The **multicast** keyword specifies IPv6 multicast address prefixes.

	Command or Action	Purpose
Step 5	**network** {*network-number* [**mask** *network-mask*] | *nsap-prefix*} [**route-map** *map-tag*] **Example:** `Device(config-router-af)# network 2001:DB8::/24`	Advertises (injects) the specified prefix into the IPv6 BGP database. (The routes must first be found in the IPv6 unicast routing table.) • Specifically, the prefix is injected into the database for the address family specified in the previous step. • Routes are tagged from the specified prefix as "local origin." • The *ipv6-prefix*argument in the **network** command must be in the form documented in RFC 2373 where the address is specified in hexadecimal using 16-bit values between colons. • The *prefix-length* argument 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 mark must precede the decimal value.
Step 6	**exit** **Example:** `Device(config-router-af)# exit`	Exits address family configuration mode, and returns the router to router configuration mode. • Repeat this step to exit router configuration mode and return the router to global configuration mode.

• What to Do Next, page 379

What to Do Next

Refer to the section "Advertising Routes into IPv6 Multiprotocol BGP" in the Implementing Multiprotocol BGP for IPv6 module for more information on assigning options to peer groups and making a BGP or multicast BGP neighbor a member of a peer group.

Redistributing Prefixes into IPv6 Multiprotocol BGP

Redistribution is the process of redistributing, or injecting, prefixes from one routing protocol into another routing protocol. This task explains how to inject prefixes from a routing protocol into IPv6 multiprotocol BGP. Specifically, prefixes that are redistributed into IPv6 multiprotocol BGP using the **redistribute** router configuration command are injected into the IPv6 unicast database.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **router bgp** *as-number*
4. **address-family ipv6** [**vrf** *vrf-name*] [**unicast** | **multicast** | **vpnv6**]
5. **redistribute bgp** [*process-id*] [**metric** *metric-value*] [**route-map** *map-name*] [*source-protocol-options*]
6. **exit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**router bgp** *as-number* **Example:** `Device(config)# router bgp 65000`	Enters router configuration mode for the specified BGP routing process.
Step 4	**address-family ipv6** [**vrf** *vrf-name*] [**unicast** | **multicast** | **vpnv6**] **Example:** `Device(config-router)# address-family ipv6`	Specifies the IPv6 address family, and enters address family configuration mode. • The **unicast** keyword specifies the IPv6 unicast address family. By default, the router is placed in configuration mode for the IPv6 unicast address family if a keyword is not specified with the **address-family ipv6** command. • The **multicast** keyword specifies IPv6 multicast address prefixes.
Step 5	**redistribute bgp** [*process-id*] [**metric** *metric-value*] [**route-map** *map-name*] [*source-protocol-options*] **Example:** `Device(config-router-af)# redistribute bgp 64500 metric 5 metric-type external`	Redistributes IPv6 routes from one routing domain into another routing domain.
Step 6	**exit** **Example:** `Device(config-router-af)# exit`	Exits address family configuration mode, and returns the router to router configuration mode. • Repeat this step to exit router configuration mode and return the router to global configuration mode.

• What to Do Next, page 380

What to Do Next

Refer to the section "Redistributing Prefixes into IPv6 Multiprotocol BGP" in the Implementing Multiprotocol BGP for IPv6 module for more information on assigning options to peer groups and making a BGP or multicast BGP neighbor a member of a peer group.

To configure aggregate addresses for Multicast BGP, refer to the "Configuring Aggregate Addresses" section of the "Configuring BGP" chapter in the *Cisco IOS XE IP Routing Configuration Guide* .

Assigning a BGP Administrative Distance

Caution Changing the administrative distance of BGP internal routes is not recommended. One problem that can occur is the accumulation of routing table inconsistencies, which can break routing.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **router bgp** *as-number*
4. **address-family ipv6** [**unicast** | **multicast**}
5. **distance bgp** *external-distance internal-distance local-distance*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**router bgp** *as-number* **Example:** `Device(config)# router bgp 100`	Enters router configuration mode for the specified routing process.
Step 4	**address-family ipv6** [**unicast**	**multicast**} **Example:** `Device(config-router)# address-family ipv6 multicast`	Enters address family configuration mode for configuring routing sessions such as BGP that use standard IPv6 address prefixes.

Command or Action	Purpose
Step 5 **distance bgp** *external-distance internal-distance local-distance* **Example:** `Device(config-router)# distance bgp 20 20 200`	Assigns a BGP administrative distance.

Generating Translate Updates for IPv6 Multicast BGP

The multicast BGP translate-update feature generally is used in a multicast BGP-capable router that peers with a customer site that has only a BGP-capable router; the customer site has not or cannot upgrade its router to a multicast BGP-capable image. Because the customer site cannot originate multicast BGP advertisements, the router with which it peers will translate the BGP prefixes into multicast BGP prefixes, which are used for multicast-source RPF lookup.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **router bgp** *as-number*
4. **address-family ipv6** [**unicast** | **multicast**}
5. **neighbor** *ipv6-address* **translate-update ipv6 multicast** [**unicast**]

DETAILED STEPS

Command or Action	Purpose
Step 1 **enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2 **configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3 **router bgp** *as-number* **Example:** `Device(config)# router bgp 100`	Enters router configuration mode for the specified routing process.

	Command or Action	Purpose
Step 4	**address-family ipv6** [**unicast** | **multicast**] **Example:** `Device(config-router)# address-family ipv6 multicast`	Enters address family configuration mode for configuring routing sessions such as BGP that use standard IPv6 address prefixes.
Step 5	**neighbor** *ipv6-address* **translate-update ipv6 multicast** [**unicast**] **Example:** `Device(config-router)# neighbor 2001:DB8:7000::2 translate-update ipv6 multicast`	Generates multiprotocol IPv6 BGP updates that correspond to unicast IPv6 updates received from a peer.

Resetting IPv6 BGP Sessions

SUMMARY STEPS

1. **enable**
2. **clear bgp ipv6** {**unicast** | **multicast**} {***** | *autonomous-system-number* | *ip-address* | *ipv6-address* | **peer-group** *peer-group-name*} [**soft**] [**in** | **out**]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**clear bgp ipv6** {**unicast** | **multicast**} {***** | *autonomous-system-number* | *ip-address* | *ipv6-address* | **peer-group** *peer-group-name*} [**soft**] [**in** | **out**] **Example:** `Device# clear bgp ipv6 unicast peer-group marketing soft out`	Resets IPv6 BGP sessions.

Clearing External BGP Peers

SUMMARY STEPS

1. **enable**
2. **clear bgp ipv6** {**unicast** | **multicast**} **external** [**soft**] [**in** | **out**]
3. **clear bgp ipv6** {**unicast** | **multicast**} **peer-group** *name*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**clear bgp ipv6** {**unicast**	**multicast**} **external** [**soft**] [**in**	**out**] **Example:** `Device# clear bgp ipv6 unicast external soft in`	Clears external IPv6 BGP peers.
Step 3	**clear bgp ipv6** {**unicast**	**multicast**} **peer-group** *name* **Example:** `Device# clear bgp ipv6 unicast peer-group marketing`	Clears all members of an IPv6 BGP peer group.

Clearing IPv6 BGP Route Dampening Information

SUMMARY STEPS

1. **enable**
2. **clear bgp ipv6** {**unicast** | **multicast**} **dampening** [*ipv6-prefix prefix-length*]

DETAILED STEPS

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

	Command or Action	Purpose
Step 2	**clear bgp ipv6** {**unicast** | **multicast**} **dampening** [*ipv6-prefix prefix-length*] **Example:** `Device# clear bgp ipv6 unicast dampening 2001:DB8::/64`	Clears IPv6 BGP route dampening information and unsuppresses the suppressed routes.

Clearing IPv6 BGP Flap Statistics

SUMMARY STEPS

1. **enable**
2. **clear bgp ipv6** {**unicast** | **multicast**} **flap-statistics** [*ipv6-prefix*/*prefix-length* | **regexp** *regexp* | **filter-list** *list*]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**clear bgp ipv6** {**unicast** | **multicast**} **flap-statistics** [*ipv6-prefix*/*prefix-length* | **regexp** *regexp* | **filter-list** *list*] **Example:** `Device# clear bgp ipv6 unicast flap-statistics filter-list 3`	Clears IPv6 BGP flap statistics.

Configuring Bandwidth-Based CAC for IPv6

Configuring the Interface Limit for Bandwidth-Based CAC in IPv6

Bandwidth-based CAC for IPv6 counts per-interface IPv6 mroute states using cost multipliers. With this feature, router administrators can specify which cost multiplier to use when accounting such state against the interface limits.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type number*
4. **ipv6 address** {*ipv6-address* / *prefix-length* | *prefix-name sub-bits*/*prefix-length*
5. **ipv6 multicast limit** [**connected** | **rpf** | **out**] *limit-acl max* [**threshold** *threshold-value*]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**interface** *type number* **Example:** `Router(config)# interface GigabitEthernet 1/3/1`	Specifies an interface type and number, and places the router in interface configuration mode.
Step 4	**ipv6 address** {*ipv6-address* / *prefix-length* | *prefix-name sub-bits*/*prefix-length* **Example:** `Router(config-if)# ipv6 address FE80::40:1:3 link-local`	Configures an IPv6 address based on an IPv6 general prefix.
Step 5	**ipv6 multicast limit** [**connected** | **rpf** | **out**] *limit-acl max* [**threshold** *threshold-value*] **Example:** `Router (config-if)# ipv6 multicast limit out acl1 10`	Configures per-interface mroute state limiters in IPv6.

Configuring an Access List for Bandwidth-Based CAC in IPv6

In bandwidth-based CAC for IPv6, router administrators can configure global limit cost commands for state matching access lists. Perform this task to configure an access list to configure a state matching access list.

or

deny

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ipv6 access-list** *access-list-name*
4. **permit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**ipv6 access-list** *access-list-name* **Example:** `Router(config)# ipv6 access-list costlist1`	Defines an IPv6 access list and places the router in IPv6 access list configuration mode.

	Command or Action	Purpose

Step 4	**permit**
	
	Example:
	
	Example:
	`or`
	
	Example:
	
	Example:
	
	deny
	
	Example:
	`Router(config-ipv6-acl)# permit any ff03::1/64`
		Use the **permit** or **deny** command to set conditions for an IPv6 access list.

Configuring the Global Limit for Bandwidth-Based CAC in IPv6

Router administrators can configure global limit cost commands for state matching access lists.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ipv6 multicast** [**vrf** *vrf-name*] **limit cost** *access-list cost-multiplier*

DETAILED STEPS

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

	Command or Action	Purpose
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**ipv6 multicast** [**vrf** *vrf-name*] **limit cost** *access-list cost-multiplier* **Example:** `Router (config)# ipv6 multicast limit cost costlist1 2`	Applies a cost to mroutes that match per-interface mroute state limiters in IPv6.

Configuring the Threshold Notification for the mCAC Limit in IPv6

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ipv6 multicast limit rate** *rate-value*
4. **interface** *type number*
5. **ipv6 multicast limit** [**connected** | **rpf** | **out**] *limit-acl max* [**threshold** *threshold-value*]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**ipv6 multicast limit rate** *rate-value* **Example:** `Router(config)# ipv6 multicast limit rate 2`	Configures the maximum allowed state on the source router.

	Command or Action	Purpose
Step 4	**interface** *type number* **Example:** `Router(config)# interface GigabitEthernet 1/3/1`	Specifies an interface type and number, and places the router in interface configuration mode.
Step 5	**ipv6 multicast limit** [**connected** | **rpf** | **out**] *limit-acl max* [**threshold** *threshold-value*] **Example:** `Router (config-if)# ipv6 multicast limit out acl1 10 threshold 20`	Configures per-interface mroute state limiters in IPv6.

Using MFIB in IPv6 Multicast

Multicast forwarding is automatically enabled when IPv6 multicast routing is enabled.

Verifying MFIB Operation in IPv6 Multicast

Note Multicast forwarding is automatically enabled when IPv6 multicast routing is enabled.

SUMMARY STEPS

1. **enable**
2. **show ipv6 mfib** [**vrf** *vrf-name*] [**link-local** | **verbose** | *group-address-name* | *ipv6-prefix / prefix-length* | *source-address-name*| **active** | **count** | **interface** | **status** | **summary**]
3. **show ipv6 mfib** [**vrf** *vrf-name*] [**link-local**| *group-name* | *group-address*] **active** [*kbps*]
4. **show ipv6 mfib** [**vrf** *vrf-name*] [**all** | **linkscope**| *group-name* | *group-address* [*source-name* | *source-address*]] **count**
5. **show ipv6 mfib interface**
6. **show ipv6 mfib status**
7. **show ipv6 mfib** [**vrf** *vrf-name*] **summary**
8. **debug ipv6 mfib** [**vrf** *vrf-name*] [*group-name*| *group-address*] [**adjacency** | **db** | **fs** | **init** | **interface** | **mrib** [**detail**] | **nat** | **pak** | **platform** | **ppr** | **ps** | **signal** | **table**]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**show ipv6 mfib** [**vrf** *vrf-name*] [**link-local** | **verbose** | *group-address-name* | *ipv6-prefix* / *prefix-length* | *source-address-name*| **active** | **count** | **interface** | **status** | **summary**] **Example:** `Device# show ipv6 mfib`	Displays the forwarding entries and interfaces in the IPv6 MFIB.
Step 3	**show ipv6 mfib** [**vrf** *vrf-name*] [**link-local**| *group-name* | *group-address*] **active** [*kbps*] **Example:** `Device# show ipv6 mfib active`	Displays the rate at which active sources are sending to multicast groups.
Step 4	**show ipv6 mfib** [**vrf** *vrf-name*] [**all** | **linkscope**| *group-name* | *group-address* [*source-name* | *source-address*]] **count** **Example:** `Device# show ipv6 mfib count`	Displays summary traffic statistics from the MFIB about the group and source.
Step 5	**show ipv6 mfib interface** **Example:** `Device# show ipv6 mfib interface`	Displays information about IPv6 multicast-enabled interfaces and their forwarding status.
Step 6	**show ipv6 mfib status** **Example:** `Device# show ipv6 mfib status`	Displays general MFIB configuration and operational status.

	Command or Action	Purpose
Step 7	**show ipv6 mfib** [**vrf** *vrf-name*] **summary** **Example:** `Device# show ipv6 mfib summary`	Displays summary information about the number of IPv6 MFIB entries and interfaces.
Step 8	**debug ipv6 mfib** [**vrf** *vrf-name*] [*group-name*| *group-address*] [**adjacency** | **db** | **fs** | **init** | **interface** | **mrib** [**detail**] | **nat** | **pak** | **platform** | **ppr** | **ps** | **signal** | **table**] **Example:** `Device# debug ipv6 mfib FF04::10 pak`	Enables debugging output on the IPv6 MFIB.

Resetting MFIB Traffic Counters

SUMMARY STEPS

1. **enable**
2. **clear ipv6 mfib** [**vrf** *vrf-name*] **counters** [*group-name* | *group-address* [*source-address* | *source-name*]]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**clear ipv6 mfib** [**vrf** *vrf-name*] **counters** [*group-name* | *group-address* [*source-address* | *source-name*]] **Example:** `Device# clear ipv6 mfib counters FF04::10`	Resets all active MFIB traffic counters.

Disabling Default Features in IPv6 Multicast

Several features are automatically enabled when IPv6 multicast is used. However, a user may want to disable certain features in response to certain situations.

- Disabling Embedded RP Support in IPv6 PIM, page 393
- Turning Off IPv6 PIM on a Specified Interface, page 394

Disabling Embedded RP Support in IPv6 PIM

A user might want to disable embedded RP support on an interface if all of the routers in the domain do not support embedded RP.

Note This task disables PIM completely, not just embedded RP support in IPv6 PIM.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **no ipv6 pim** [**vrf** *vrf-name*] **rp embedded**
4. **interface** *type number*
5. **no ipv6 pim**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**no ipv6 pim** [**vrf** *vrf-name*] **rp embedded** **Example:** `Router(config)# no ipv6 pim rp embedded`	Disables embedded RP support in IPv6 PIM.
Step 4	**interface** *type number* **Example:** `Router(config)# interface GigabitEthernet 1/0/0`	Specifies an interface type and number, and places the router in interface configuration mode.

	Command or Action	Purpose
Step 5	**no ipv6 pim** **Example:** `Router(config-if)# no ipv6 pim`	Turns off IPv6 PIM on a specified interface.

Turning Off IPv6 PIM on a Specified Interface

A user might only want specified interfaces to perform IPv6 multicast and will therefore want to turn off PIM on a specified interface.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type number*
4. **no ipv6 pim**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**interface** *type number* **Example:** `Router(config)# interface GigabitEthernet 1/0/0`	Specifies an interface type and number, and places the router in interface configuration mode.
Step 4	**no ipv6 pim** **Example:** `Router(config-if)# no ipv6 pim`	Turns off IPv6 PIM on a specified interface.

Disabling MLD Router-Side Processing

A user might only want specified interfaces to perform IPv6 multicast and will therefore want to turn off MLD router-side processing on a specified interface.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type number*
4. **no ipv6 mld router**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**interface** *type number* **Example:** `Router(config)# interface GigabitEthernet 1/0/0`	Specifies an interface type and number, and places the router in interface configuration mode.
Step 4	**no ipv6 mld router** **Example:** `Router(config-if)# no ipv6 mld router`	Disables MLD router-side processing on a specified interface.

Disabling MFIB on the device

Multicast forwarding is automatically enabled when IPv6 multicast routing is enabled. However, a user may want to disable multicast forwarding on the router.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **no ipv6 mfib**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**no ipv6 mfib** **Example:** `Device(config)# no ipv6 mfib`	Disables IPv6 multicast forwarding on the device.

Disabling MFIB Interrupt-Level IPv6 Multicast Forwarding

MFIB interrupt-level IPv6 multicast forwarding of outgoing packets on a specific interface is enabled on interfaces that support Cisco Express Forwarding. However, a user may want to disable MFIB interrupt-level forwarding on a specified interface.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type number*
4. **no ipv6 mfib cef output**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**interface** *type number* **Example:** `Router(config)# interface GigabitEthernet 1/0/0`	Specifies an interface type and number, and places the router in interface configuration mode.
Step 4	**no ipv6 mfib cef output** **Example:** `Router(config-if)# no ipv6 mfib cef output`	Disables MFIB interrupt-level IPv6 multicast forwarding of outgoing packets on a specific interface.

Troubleshooting IPv6 Multicast

SUMMARY STEPS

1. **enable**
2. **debug ipv6 mfib** *group-name* | *group-address*] [**adjacency** | **signal** | **db** | **init** | **mrib** | **pak** | **ps**
3. debug ipv6 mld [group-name | group-address | interface-type]
4. **debug ipv6 mld explicit** [*group-name* | *group-address*
5. **debug ipv6 pim** [*group-name* | *group-address* | *interface-type* | **neighbor** | **bsr**
6. **debug bgp ipv6** {**unicast** | **multicast**} **dampening** [**prefix-list** *prefix-list-name*
7. **debug bgp ipv6** {**unicast** | **multicast**} **updates** [*ipv6-address*] [**prefix-list** *prefix-list-name*] [**in** | **out**
8. **debug ipv6 mrib client**
9. **debug ipv6 mrib io**
10. **debug ipv6 mrib issu**
11. **debug ipv6 mrib proxy**
12. **debug ipv6 mrib route** [*group-name* | *group-address*
13. **debug ipv6 mrib table**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**debug ipv6 mfib** *group-name* | *group-address*] [**adjacency** | **signal** | **db** | **init** | **mrib** | **pak** | **ps** **Example:** `Router# debug ipv6 mfib pak FF04::10`	Enables debugging output on the IPv6 MFIB.
Step 3	debug ipv6 mld [group-name | group-address | interface-type] **Example:** `Router# debug ipv6 mld`	Enables debugging on MLD protocol activity.
Step 4	**debug ipv6 mld explicit** [*group-name* | *group-address* **Example:** `Router# debug ipv6 mld explicit`	Displays information related to the explicit tracking of hosts.
Step 5	**debug ipv6 pim** [*group-name* | *group-address* | *interface-type* | **neighbor** | **bsr** **Example:** `Router# debug ipv6 pim`	Enables debugging on PIM protocol activity.
Step 6	**debug bgp ipv6** {**unicast** | **multicast**} **dampening** [**prefix-list** *prefix-list-name* **Example:** `Router# debug bgp ipv6 multicast`	Displays debugging messages for IPv6 BGP dampening.

	Command or Action	Purpose
Step 7	**debug bgp ipv6** {**unicast** | **multicast**} **updates** [*ipv6-address*] [**prefix-list** *prefix-list-name*] [**in** | **out** **Example:** `Router# debug bgp ipv6 multicast updates`	Displays debugging messages for IPv6 BGP update packets.
Step 8	**debug ipv6 mrib client** **Example:** `Router# debug ipv6 mrib client`	Enables debugging on MRIB client management activity.
Step 9	**debug ipv6 mrib io** **Example:** `Router# debug ipv6 mrib io`	Enables debugging on MRIB I/O events.
Step 10	**debug ipv6 mrib issu** **Example:** `Router# debug ipv6 mrib issu`	Enables debugging on MRIB in service software update.
Step 11	**debug ipv6 mrib proxy** **Example:** `Router# debug ipv6 mrib proxy`	Enables debugging on MRIB proxy activity between the route processor and line cards on distributed router platforms.
Step 12	**debug ipv6 mrib route** [*group-name* | *group-address* **Example:** `Router# debug ipv6 mrib route`	Displays information about MRIB routing entry-related activity.
Step 13	**debug ipv6 mrib table** **Example:** `Router# debug ipv6 mrib table`	Enables debugging on MRIB table management activity.

- Examples, page 400

Examples

Sample Output from the show ipv6 mfib Command

The following example displays the forwarding entries and interfaces in the MFIB. The router is configured for fast switching, and it has a receiver joined to FF05::1 on GigabitEthernet 1/1/0 and a source (2001:DB8:1:1:20) sending on GigabitEthernet 1/2/0:

```
Router# show ipv6 mfib
IP Multicast Forwarding Information Base
Entry Flags: C - Directly Connected, S - Signal, IA - Inherit A flag,
             AR - Activity Required, D - Drop
Forwarding Counts: Pkt Count/Pkts per second/Avg Pkt Size/Kbits per second
Other counts: Total/RPF failed/Other drops
Interface Flags: A - Accept, F - Forward, NS - Negate Signalling
             IC - Internal Copy, NP - Not platform switched
             SP - Signal Present
Interface Counts: FS Pkt Count/PS Pkt Count
(*,FF00::/8) Flags: C
   Forwarding: 0/0/0/0, Other: 0/0/0
   Tunnel0 Flags: NS
(*,FF00::/15) Flags: D
   Forwarding: 0/0/0/0, Other: 0/0/0
(*,FF05::1) Flags: C
   Forwarding: 2/0/100/0, Other: 0/0/0
   Tunnel0 Flags: A NS
   GigabitEthernet1/1/0 Flags: F NS
     Pkts: 0/2
(2001:DB8:1:1:200,FF05::1) Flags:
   Forwarding: 5/0/100/0, Other: 0/0/0
   GigabitEthernet1/2/0 Flags: A
   GigabitEthernet1/1/0 Flags: F NS
     Pkts: 3/2
(*,FF10::/15) Flags: D
   Forwarding: 0/0/0/0, Other: 0/0/0
```

Sample Output from the show ipv6 mfib active Command

The following example displays statistics on the rate at which active IP multicast sources are sending information. The router is switching traffic from 2001:DB8:1:1:200 to FF05::1:

```
Router# show ipv6 mfib active
Active IPv6 Multicast Sources - sending >= 4 kbps
Group: FF05::1
  Source: 2001:DB8:1:1:200
    Rate: 20 pps/16 kbps(1sec), 0 kbps(last 128 sec)
```

Sample Output from the show ipv6 mfib count Command

The following example displays statistics from the MFIB about the group and source. The router is switching traffic from 2001:DB8:1:1:200 to FF05::1:

```
Router# show ipv6 mfib count

IP Multicast Statistics
54 routes, 7 groups, 0.14 average sources per group
Forwarding Counts: Pkt Count/Pkts per second/Avg Pkt Size/Kilobits per second
Other counts: Total/RPF failed/Other drops(OIF-null, rate-limit etc)
Group: FF00::/8
  RP-tree:    Forwarding: 0/0/0/0, Other: 0/0/0
Group: FF00::/15
  RP-tree:    Forwarding: 0/0/0/0, Other: 0/0/0
Group: FF05::1
  RP-tree:    Forwarding: 2/0/100/0, Other: 0/0/0
```

```
  Source: 10::1:1:200,   Forwarding: 367/10/100/7, Other: 0/0/0
  Tot. shown: Source count: 1, pkt count: 369
Group: FF10::/15
  RP-tree:    Forwarding: 0/0/0/0, Other: 0/0/0
Group: FF20::/15
  RP-tree:    Forwarding: 0/0/0/0, Other: 0/0/0
```

Sample Output from the show ipv6 mfib interface Command

The following example displays information about IPv6 multicast-enabled interfaces and their forwarding status. The router is configured for fast switching:

```
Router# show ipv6 mfib interface
IPv6 Multicast Forwarding (MFIB) status:
    Configuration Status: enabled
    Operational Status: running
MFIB interface       status    CEF-based output
                               [configured,available]
GigabitEthernet1/1/0 up     [yes       ,yes      ]
GigabitEthernet1/2/0 up     [yes       ,?        ]
Tunnel0              up     [yes       ,?        ]
Tunnel1              up     [yes       ,?        ]
```

Sample Output from the show ipv6 mfib summary Command

The following example displays summary information about the number of IPv6 MFIB entries and interfaces:

```
Router# show ipv6 mfib summary

IPv6 MFIB summary:
  54     total entries [1 (S,G), 7 (*,G), 46 (*,G/m)]
  17     total MFIB interfaces
```

Sample Output from the show ipv6 mld groups Command

The following is sample output from the **show ipv6 mld groups** command. It shows all of the groups joined by Gigabit Ethernet interface 2/1/0, including link-local groups used by network protocols.

```
Router# show ipv6 mld groups GigabitEthernet 2/1/0
MLD Connected Group Membership
Group Address          Interface           Uptime        Expires
FF02::2                GigabitEthernet2/1/0     3d18h         never
FF02::D                GigabitEthernet2/1/0     3d18h         never
FF02::16               GigabitEthernet2/1/0     3d18h         never
FF02::1:FF00:1         GigabitEthernet2/1/0     3d18h         00:00:27
FF02::1:FF00:79        GigabitEthernet2/1/0     3d18h         never
FF02::1:FF23:83C2      GigabitEthernet2/1/0     3d18h         00:00:22
FF02::1:FFAF:2C39      GigabitEthernet2/1/0     3d18h         never
FF06:7777::1           GigabitEthernet2/1/0     3d18h         00:00:26
```

Sample Output from the show ipv6 mld groups summary Command

The following is sample output from the **show ipv6 mld groups summary**command:

```
Router# show ipv6 mld groups summary
MLD Route Summary
  No. of (*,G) routes = 5
  No. of (S,G) routes = 0
```

Sample Output from the show ipv6 mld interface Command

The following is sample output from the **show ipv6 mld interface** command for Gigabit Ethernet interface 2/1/0:

```
Router# show ipv6 mld interface GigabitEthernet 2/1/0
GigabitEthernet2/1/0 is up, line protocol is up
Internet address is FE80::205:5FFF:FEAF:2C39/10
MLD is enabled in interface
Current MLD version is 2
MLD query interval is 125 seconds
MLD querier timeout is 255 seconds
MLD max query response time is 10 seconds
Last member query response interval is 1 seconds
MLD activity: 25 joins, 17 leaves
MLD querying router is FE80::205:5FFF:FEAF:2C39 (this system)
```

Sample Output from the show ipv6 mld ssm-map Command

The following examples show SSM mapping for the source address 2001:DB8::1:

```
Router# show ipv6 mld ssm-map 2001:DB8::1
 Group address  : 2001:DB8::1
 Group mode ssm : TRUE
 Database       : STATIC
 Source list    : 2001:DB8::2
                  2001:DB8::3
Router# show ipv6 mld ssm-map 2001:DB8::2
 Group address  : 2001:DB8::2
 Group mode ssm : TRUE
 Database       : DNS
 Source list    : 2001:DB8::3
                  2001:DB8::1
```

Sample Output from the show ipv6 mld traffic Command

The following example displays the MLD protocol messages received and sent.

```
Router# show ipv6 mld traffic

MLD Traffic Counters
Elapsed time since counters cleared:00:00:21
                              Received     Sent
Valid MLD Packets               3           1
Queries                         1           0
Reports                         2           1
Leaves                          0           0
Mtrace packets                  0           0
Errors:
Malformed Packets                           0
Bad Checksums                               0
Martian source                              0
Packets Received on MLD-disabled Interface  0
```

Sample Output from the show ipv6 mrib client Command

The following is sample output from the **show ipv6 mrib client**command:

```
Router# show ipv6 mrib client
IP MRIB client-connections
igmp:145        (connection id 0)
pim:146 (connection id 1)
mfib ipv6:3     (connection id 2)
slot 3  mfib ipv6 rp agent:16   (connection id 3)
slot 1  mfib ipv6 rp agent:16   (connection id 4)
slot 0  mfib ipv6 rp agent:16   (connection id 5)
```

```
slot 4  mfib ipv6 rp agent:16   (connection id 6)
slot 2  mfib ipv6 rp agent:16   (connection id 7)
```

Sample Output from the show ipv6 mrib route Command

The following is sample output from the **show ipv6 mrib route**command using the **summary** keyword:

```
Router# show ipv6 mrib route summary
MRIB Route-DB Summary
  No. of (*,G) routes = 52
  No. of (S,G) routes = 0
  No. of Route x Interfaces (RxI) = 10
```

Sample Output from the show ipv6 mroute Command

Using the **show ipv6 mroute** command is a good way to dynamically verify that multicast IPv6 data is flowing. The following is sample output from the **show ipv6 mroute**command:

```
Router# show ipv6 mroute ff07::1
Multicast Routing Table
Flags:D - Dense, S - Sparse, B - Bidir Group, s - SSM Group,
       C - Connected, L - Local, I - Received Source Specific Host Report,
       P - Pruned, R - RP-bit set, F - Register flag, T - SPT-bit set,
       J - Join SPT
Timers:Uptime/Expires
Interface state:Interface, State
(*, FF07::1), 00:04:45/00:02:47, RP 2001:DB8:6::6, flags:S
  Incoming interface:Tunnel5
  RPF nbr:6:6:6::6
  Outgoing interface list:
    POS4/0, Forward, 00:04:45/00:02:47
(2001:DB8:999::99, FF07::1), 00:02:06/00:01:23, flags:SFT
  Incoming interface:POS1/0
  RPF nbr:2001:DB8:999::99
  Outgoing interface list:
    POS4/0, Forward, 00:02:06/00:03:27
```

Sample Output from the show ipv6 mroute active Command

The following is sample output from the **show ipv6 mroute active**command:

```
Router# show ipv6 mroute active
Active IPv6 Multicast Sources - sending >= 4 kbps
Group:FF05::1
 Source:2001:DB8:1:1:1
   Rate:11 pps/8 kbps(1sec), 8 kbps(last 8 sec)
```

Sample Output from the show ipv6 pim group-map Command

The following is sample output from the **show ipv6 pim group-map**command:

```
Router# show ipv6 pim group-map
FF33::/32*
      SSM
      Info source:Static
      Uptime:00:08:32, Groups:0
  FF34::/32*
      SSM
      Info source:Static
      Uptime:00:09:42, Groups:0
```

Sample Output from the show ipv6 pim interface Command

The following is sample output from the **show ipv6 pim interface** command using the **state-on** keyword:

```
Router# show ipv6 pim interface state-on
Interface          PIM  Nbr   Hello  DR
                        Count Intvl  Prior
GigabitEthernet0/0/0 on   0     30     1
    Address:FE80::208:20FF:FE08:D7FF
    DR     :this system
POS1/0             on   0     30     1
    Address:FE80::208:20FF:FE08:D554
    DR     :this system
POS4/0             on   1     30   1
    Address:FE80::208:20FF:FE08:D554
    DR     :FE80::250:E2FF:FE8B:4C80
POS4/1             on   0     30   1
    Address:FE80::208:20FF:FE08:D554
    DR     :this system
Loopback0          on   0     30     1
    Address:FE80::208:20FF:FE08:D554
    DR     :this system
```

Sample Output from the show ipv6 pim join-prune statistic Command

The following example provides the join/prune aggregation on GigabitEthernet interface 0/0/0:

```
Router# show ipv6 pim join-prune statistic GigabitEthernet0/0/0
PIM Average Join/Prune Aggregation for last (1K/10K/50K) packets
Interface             Transmitted             Received
GigabitEthernet0/0/0      0    / 0    / 0         1    / 0    / 0
```

Sample Output from the show ipv6 pim range-list Command

The following is sample output from the **show ipv6 pim range-list**command:

```
Router# show ipv6 pim range-list
config SSM Exp:never Learnt from :::
 FF33::/32 Up:00:26:33
 FF34::/32 Up:00:26:33
 FF35::/32 Up:00:26:33
 FF36::/32 Up:00:26:33
 FF37::/32 Up:00:26:33
 FF38::/32 Up:00:26:33
 FF39::/32 Up:00:26:33
 FF3A::/32 Up:00:26:33
 FF3B::/32 Up:00:26:33
 FF3C::/32 Up:00:26:33
 FF3D::/32 Up:00:26:33
 FF3E::/32 Up:00:26:33
 FF3F::/32 Up:00:26:33
config SM RP:40::1:1:1 Exp:never Learnt from :::
 FF13::/64 Up:00:03:50
config SM RP:40::1:1:3 Exp:never Learnt from :::
 FF09::/64 Up:00:03:50
```

Sample Output from the show ipv6 pim topology Command

The following is sample output from the **show ipv6 pim topology** command:

```
Router# show ipv6 pim topology
IP PIM Multicast Topology Table
Entry state:(*/S,G)[RPT/SPT] Protocol Uptime Info
Entry flags:KAT - Keep Alive Timer, AA - Assume Alive, PA - Probe Alive,
    RA - Really Alive, LH - Last Hop, DSS - Don't Signal Sources,
    RR - Register Received, SR - Sending Registers, E - MSDP External,
    DCC - Don't Check Connected
```

```
Interface state:Name, Uptime, Fwd, Info
Interface flags:LI - Local Interest, LD - Local Dissinterest,
II - Internal Interest, ID - Internal Dissinterest,
LH - Last Hop, AS - Assert, AB - Admin Boundary
(*,FF05::1)
SM UP:02:26:56 JP:Join(now) Flags:LH
RP:2001:DB8:1:1:2
RPF:GigabitEthernet1/1/0,FE81::1
  GigabitEthernet0/1/0 02:26:56  fwd LI LH
(2001:DB8:1:1:200,FF05::1)
SM UP:00:00:07 JP:Null(never) Flags:
RPF:GigabitEthernet1/1/0,FE80::30:1:4
  GigabitEthernet1/1/0         00:00:07  off LI
```

Sample Output from the show ipv6 pim traffic Command

The following example shows the number of PIM protocol messages received and sent.

```
Router# show ipv6 pim traffic

PIM Traffic Counters
Elapsed time since counters cleared:00:05:29
                              Received     Sent
Valid PIM Packets               22          22
Hello                           22          22
Join-Prune                      0           0
Register                        0           0
Register Stop                   0           0
Assert                          0           0
Bidir DF Election               0           0
Errors:
Malformed Packets                           0
Bad Checksums                               0
Send Errors                                 0
Packet Sent on Loopback Errors              0
Packets Received on PIM-disabled Interface  0
Packets Received with Unknown PIM Version   0
```

Sample Output from the show ipv6 pim tunnel Command

The following is sample output from the **show ipv6 pim tunnel**command on the RP:

```
Router# show ipv6 pim tunnel
Tunnel0*
 Type  :PIM Encap
 RP    :100::1
 Source:100::1
Tunnel0*
 Type  :PIM Decap
 RP    :100::1
 Source: -
```

The following is sample output from the **show ipv6 pim tunnel**command on a non-RP:

```
Router# show ipv6 pim tunnel
Tunnel0*
 Type  :PIM Encap
 RP    :100::1
 Source:2001::1:1:1
```

Sample Output from the show ipv6 rpf Command

The following example displays RPF information for the unicast host with the IPv6 address of 2001:DB8:1:1:2:

```
Router# show ipv6 rpf 2001:DB8:1:1:2
RPF information for 2001:DB8:1:1:2
```

```
RPF interface:GigabitEthernet3/2/0
RPF neighbor:FE80::40:1:3
RPF route/mask:20::/64
RPF type:Unicast
RPF recursion count:0
Metric preference:110
Metric:30
```

Configuration Examples for IPv6 Multicast

Example: Enabling IPv6 Multicast Routing

The following example enables multicast routing on all interfaces. Entering this command also enables multicast forwarding for PIM and MLD on all enabled interfaces of the router.

```
Router> enable
Router# configure terminal

Router(config)# ipv6 multicast-routing
```

Examples Configuring the MLD Protocol

The following example shows how to configure the query maximum response time, the query timeout, and the query interval on GigabitEthernet interface 1/0/0:

```
Router> enable
Router# configure terminal
Router(config)# interface GigabitEthernet 1/0/0

Router(config-if)# ipv6 mld query-max-response-time 20

Router(config-if)# i pv6 mld query-timeout 130

Router(config-if)# ipv6 mld query-interval 60
```

The following example configures MLD reporting for a specified group and source, allows the user to perform IPv6 multicast receiver access control, and statically forwards traffic for the multicast group onto GigabitEthernet interface 1/0/0:

```
Router> enable
Router# configure terminal
Router(config)# interface GigabitEthernet 1/0/0
Router(config)# ipv6 mld join-group FF04::10
Router(config)# ipv6 mld static-group FF04::10 100::1
Router(config)# ipv6 mld access-group acc-grp-1
```

Example Configuring Explicit Tracking of Receivers

```
Router> enable
Router# configure terminal
Router(config)# interface GigabitEthernet 1/0/0
Router(config-if)# ipv6 mld explicit-tracking list1
```

Example Configuring PIM

The following example shows how to configure a router to use PIM-SM using 2001:DB8::1 as the RP. It sets the SPT threshold to infinity to prevent switchover to the source tree when a source starts sending traffic and sets a filter on all sources that do not have a local multicast BGP prefix.

```
Router(config)# ipv6 multicast-routing
Router(config)# ipv6 pim rp-address 2001:DB8::1
Router(config)# ipv6 pim spt-threshold infinity
```

Example Configuring PIM Options

The following example sets the DR priority, the PIM hello interval, and the periodic join and prune announcement interval on GigabitEthernet interface 0/0/0.

```
Router(config)# interface GigabitEthernet0/0/0
Router(config)# ipv6 pim hello-interval 60
Router(config)# ipv6 pim dr-priority 3
Router(config)# ipv6 pim join-prune-interval 75
```

Example Configuring Mroutes

The following example shows how to configure a static multicast route to be used for multicast RPF selection only:

```
Router> enable
Router# configure terminal
Router(config)# ipv6 route 2001:DB8::/64 7::7 100 multicast
```

Example Configuring an IPv6 Multiprotocol BGP Peer Group

The following example configures the IPv6 multiprotocol BGP peer group named group1:

```
router bgp 65000
no bgp default ipv4-unicast
neighbor group1 peer-group
neighbor 2001:DB8:0:CC00::1 remote-as 64600
address-family ipv6 multicast
 neighbor 3FFE:C00:0:1:A8BB:CCFF:FE00:8200 activate
```

```
 no auto-summary
 no synchronization
 exit-address-family
```

Example Redistributing Prefixes into IPv6 Multiprotocol BGP

The following example redistributes BGP routes into the IPv6 multicast database of the local router:

```
router bgp 64900
 no bgp default ipv4-unicast
address-family ipv6 multicast
 redistribute BGP
```

Example: Generating Translate Updates for IPv6 Multicast BGP

The following example shows how to generate IPv6 multicast BGP updates that correspond to unicast IPv6 updates:

```
router bgp 64900
 no bgp default ipv4-unicast
address-family ipv6 multicast
 neighbor 2001:DB8:7000::2 translate-update ipv6 multicast
```

Example: Configuring Bandwidth-Based CAC for IPv6

Example: Configuring the Interface Limit for Bandwidth-Based CAC in IPv6

The following example configures the interface limit on the source router's outgoing interface GigabitEthernet 1/1/3.

```
interface GigabitEthernet 1/3/1
ipv6 address FE80::40:1:3 link-local
 ipv6 address 2001:DB8:1:1:3/64
 ipv6 multicast limit out acl1 10
```

Example: Configuring an Access List for Bandwidth-Based CAC in IPv6

The following example shows how to configure an access list to use for bandwidth-based CAC:

```
ipv6 access-list cost-list
   permit any ff03::1/64
```

Example: Configuring the Global Limit for Bandwidth-Based CAC

The following example configures the global limit on the source router.

```
ipv6 multicast limit cost cost-list 2
```

Example Turning Off IPv6 PIM on a Specified Interface

The following example turns off IPv6 PIM on GigabitEthernet interface 1/0/0:

```
Router(config)# ipv6 multicast-routing
Router(config)# interface GigabitEthernet 1/0/0
Router(config)# no ipv6 pim
```

Example Disabling MLD Router-Side Processing

The following example turns off MLD router-side processing on GigabitEthernet interface 1/0/0:

```
Router> enable
Router# configure terminal
Router(config)# interface GigabitEthernet 1/0/0

Router(config-if)# no ipv6 mld router
```

Additional References

Related Documents

Related Topic	Document Title
IPv6 multicast addresses	Implementing IPv6 Addressing and Basic Connectivity , *Cisco IOS XE IPv6 Configuration Guide*
Multicast BGP for IPv6	Implementing Multiprotocol BGP for IPv6 , *Cisco IOS XE IPv6 Configuration Guide*
IPv6 static routes	Implementing Static Routes for IPv6 , *Cisco IOS XE IPv6 Configuration Guide*
IPv6 tunnels	Implementing Tunneling for IPv6 , *Cisco IOS XE IPv6 Configuration Guide*

Standards and Drafts

Standards	Title
draft-ietf-pim-sm-v2-new	*Protocol Independent Multicast - Sparse Mode PIM-SM): Protocol Specification (Revised)* , March 6, 2003
draft-savola-mboned-mcast-rpaddr	*Embedding the Address of RP in IPv6 Multicast Address* , May 23, 2003
draft-suz-pim-upstream-detection	*PIM Upstream Detection Among Multiple Addresses* , February 2003

Standards	Title
draft-ietf-pim-bidir-05	*Bi-directional Protocol Independent Multicast (BIDIR-PIM)* , June 20, 2003

MIBs

MIBs	MIBs Link
None	To locate and download MIBs for selected platforms, Cisco software releases, and feature sets, use Cisco MIB Locator found at the following URL: http://www.cisco.com/go/mibs

RFCs

RFCs	Title
RFC 2373	*IP Version 6 Addressing Architecture*
RFC 2375	*IPv6 Multicast Address Assignments*
RFC 2460	*Internet Protocol, Version 6 (IPv6) Specification*
RFC 2461	*Neighbor Discovery for IP version 6 (IPv6)*
RFC 2462	*IPv6 Stateless Address Autoconfiguration*
RFC 3576	*Change of Authorization*
RFC 3590	*Source Address Selection for the Multicast Listener Discovery (MLD) Protocol*
RFC 3810	*Multicast Listener Discovery Version 2 (MLDv2) for IPv6*
RFC 4007	*IPv6 Scoped Address Architecture*
RFC 4610	*Anycast-RP Using Protocol Independent Multicast (PIM)*

Technical Assistance

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

Feature Information for Implementing IPv6 Multicast

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Table 17 Feature Information for Implementing IPv6 Multicast

Feature Name	Releases	Feature Information
Distributed MFIB (dMFIB)	Cisco IOS XE Release 2.1	Distributed MFIB is used to switch multicast IPv6 packets on distributed platforms.
IPv6 Multicast	Cisco IOS XE Release 2.1	IPv6 multicast allows a host to send a single data stream to a subset of all hosts simultaneously.
IPv6--Multicast Address Group Range Support	Cisco IOS XE Release 2.6 Cisco IOS XE Release 3.3SG	This feature allows the router to keep from receiving multicast traffic to be received from unauthenticated groups or unauthorized channels. The following command was modified by this feature: **ipv6 multicast group-range**.
IPv6 Multicast--Address Family Support for Multiprotocol BGP	Cisco IOS XE Release 2.1	This feature provides multicast BGP extensions for IPv6 and supports the same features and functionality as IPv4 BGP.

Feature Name	Releases	Feature Information
IPv6 Multicast--Bandwidth-Based Call Admission Control (CAC)	Cisco IOS XE Release 2.6	The bandwidth-based call admission control (CAC) for IPv6 multicast feature implements a method to monitor bandwidth per interface and multicast group avoiding oversubscription due to multicast services. The following commands were modified by this feature: ipv6 multicast group-range, ipv6 multicast limit, ipv6 multicast limit cost.
IPv6 Multicast--Bootstrap Router (BSR)	Cisco IOS XE Release 2.1 Cisco IOS XE Release 2.4	If an RP becomes unreachable, this feature allows the RP to be detected and the mapping tables modified so that the unreachable RP is no longer used, and the new tables will be rapidly distributed throughout the domain.
IPv6 Multicast--Explicit Tracking of Receivers	Cisco IOS XE Release 2.1	This feature allows a router to track the behavior of the hosts within its IPv6 network. The following command was modified by this feature: **ipv6 mld explicit-tracking**
IPv6 Multicast--IPv6 Bidirectional PIM	Cisco IOS XE Release 2.3	Bidirectional PIM allows multicast routers to keep reduced state information. Bidirectional shared trees convey data from sources to the RP and distribute them from the RP to the receivers. The following commands were modified by this feature: **ipv6 pim rp-address**, **show ipv6 pim df**, **show ipv6 pim df winner**
IPv6 Multicast--IPv6 BSR--Ability to Configure RP Mapping	Cisco IOS XE Release 2.4 Cisco IOS XE Release 3.3SG	This feature allows IPv6 multicast routers to be statically configured to announce scope-to-RP mappings directly from the BSR instead of learning them from candidate-RP messages.

Feature Name	Releases	Feature Information
IPv6 Multicast--IPv6 BSR Bidirectional Support	Cisco IOS XE Release 2.4	Bidirectional BSR support allows bidirectional RPs to be advertised in C-RP messages and bidirectional ranges in the BSM.
IPv6 Multicast--MLD Access Group	Cisco IOS XE Release 2.1	The MLD access group provides receiver access control in Cisco IOS XE IPv6 multicast routers. The following command was modified by this feature: **ipv6 mld access-group**
IPv6 Multicast--MLD Group Limits	Cisco IOS XE Release 2.6 Cisco IOS XE Release 3.3SG	The MLD group limits feature provides protection against denial of service (DoS) attacks caused by MLD packets. The following commands were modified by this feature: **ipv6 mld limit**, **ipv6 mld state-limit**

Feature Name	Releases	Feature Information
IPv6 Multicast--Multicast Listener Discovery (MLD) Protocol, Versions 1 and 2	Cisco IOS XE Release 2.1	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. There are two versions of MLD: MLD version 1 is based on version 2 of the IGMP for IPv4, and MLD version 2 is based on version 3 of the IGMP for IPv4. IPv6 multicast for Cisco IOS XE software uses both MLD version 2 and MLD version 1. The following commands were modified by this feature: **clear ipv6 mld counters**, **clear ipv6 mld traffic**, **debug ipv6 mld**, **debug ipv6 mld explicit**, **debug ipv6 mld ssm-map**, **ipv6 mld join-group**, **ipv6 mld query-interval**, **ipv6 mld query-max-response-time**, **ipv6 mld query-timeout**, **ipv6 mld router**, **ipv6 mld static-group**, **ipv6 multicast-routing**, **show ipv6 mld interface**, **show ipv6 mld groups**, **show ipv6 mld groups summary**, **show ipv6 mld traffic**
IPv6 Multicast--MRIB	Cisco IOS XE Release 2.1	The MRIB is a protocol-independent repository of multicast routing entries instantiated by multicast routing protocols (routing clients). The following commands were modified by this feature: **clear ipv6 pim topology**, **debug ipv6 mrib client, debug ipv6 mrib io, debug ipv6 mrib proxy, debug ipv6 mrib route, debug ipv6 mrib table, show ipv6 mrib client**, **show ipv6 mrib route**, **show ipv6 pim topology**

Feature Name	Releases	Feature Information
IPv6 Multicast--PIM Source Specific Multicast (PIM-SSM)	Cisco IOS XE Release 2.1	PIM-SSM supports the implementation of SSM and is derived from PIM-SM. The SSM feature forwards datagram traffic to receivers from only those multicast sources that the receivers have explicitly joined, optimizing bandwidth utilization and denying unwanted Internet broadcast traffic. The following commands were modified by this feature: **clear ipv6 pim counters**, **clear ipv6 pim topology**, **debug ipv6 pim**, **debug ipv6 pim df-election**, **ipv6 pim**, **ipv6 pim dr-priority**, **ipv6 pim hello-interval**, **ipv6 pim join-prune-interval**, **ipv6 pim spt-threshold infinity**, **show ipv6 mrib client**, **show ipv6 mrib route**, **show ipv6 pim group-map**, **show ipv6 pim interface**, **show ipv6 pim join-prune statistic**, **show ipv6 pim range-list**, **show ipv6 pim traffic**, **show ipv6 pim topology**
IPv6 Multicast--PIM Sparse Mode (PIM-SM)	Cisco IOS XE Release 2.1 Cisco IOS XE Release 2.4	PIM-SM uses unicast routing to provide reverse-path information for multicast tree building. PIM-SM is used in a multicast network when relatively few routers are involved in each multicast and these routers do not forward multicast packets for a group, unless there is an explicit request for the traffic.
IPv6 Multicast--Routable Address Hello Option	Cisco IOS XE Release 2.4	The routable address hello option adds a PIM hello message option that includes all the addresses on the interface on which the PIM hello message is advertised.

Feature Name	Releases	Feature Information
IPv6 Multicast--SSM Mapping for MLDv1 SSM	Cisco IOS XE Release 2.1 Cisco IOS XE Release 2.4 Cisco IOS XE Release 3.3SG	This feature allows deployment of IPv6 SSM with hosts that are incapable of providing MLD version 2 support in their TCP/IP host stack and their IP multicast receiving application. The following commands were modified by this feature: **ipv6 mld ssm-map enable**, **ipv6 mld ssm-map query dns**, **ipv6 mld ssm-map static**, **show ipv6 mld ssm-map**
IPv6 Multicast--Static Multicast Routing (mroute)	Cisco IOS XE Release 2.1	IPv6 static mroutes share the same database as IPv6 static routes and are implemented by extending static route support. The following commands were modified by this feature: **ipv6 route**, **show ipv6 mroute**, **show ipv6 mroute active**, **show ipv6 rpf**
IPv6 Multicast--VRF Lite	XE 3.4S	The IPv6 Multicast VRF Lite feature provides IPv6 multicast support for multiple virtual routing/forwarding contexts (VRFs). The scope of these VRFs is limited to the router in which the VRFs are defined.
PIM Passive Mode	Cisco IOS XE Release 2.6	This feature allows PIM passive mode to be enabled on an interface so that a PIM passive interface cannot send and receive PIM control messages, but it can act as RPF interface for multicast route entries, and it can accept and forward multicast data packets. The following command were introduced or modified by this feature: **ipv6 multicast pim-passive-enable**, **ipv6 pim passive**.

Feature Name	Releases	Feature Information
Threshold Notification for mCAC Limit	Cisco IOS XE Release 2.6	Support for this feature is provided in Cisco IOS XE Release 2.6 The following command were introduced or modified by this feature: ipv6 multicast limit, ipv6 multicast limit rate.
PIMv6: Anycast RP Solution	Cisco IOS XE Release 3.4S	The anycast RP solution in IPv6 PIM allows an IPv6 network to support anycast services for the PIM-SM RP. It allows anycast RP to be used inside a domain that runs PIM only. This feature is useful when interdomain connection is not required.

PIMv6 Anycast RP Solution

Finding Feature Information

Your software release may not support all the features documented in this module. For the latest caveats and feature information, see Bug Search Tool and the release notes for your platform and software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the feature information table at the end of this module.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Information About the PIMv6 Anycast RP Solution

PIMv6 Anycast RP Solution Overview

The anycast RP solution in IPv6 PIM allows an IPv6 network to support anycast services for the PIM-SM RP. It allows anycast RP to be used inside a domain that runs PIM only. Anycast RP can be used in IPv4 as well as IPv6, but it does not depend on the Multicast Source Discovery Protocol (MSDP), which runs only on IPv4. This feature is useful when interdomain connection is not required.

Anycast RP is a mechanism that ISP-based backbones use to get fast convergence when a PIM RP device fails. To allow receivers and sources to rendezvous to the closest RP, the packets from a source need to get to all RPs to find joined receivers.

A unicast IP address is chosen as the RP address. This address is either statically configured or distributed using a dynamic protocol to all PIM devices throughout the domain. A set of devices in the domain is chosen to act as RPs for this RP address; these devices are called the anycast RP set. Each device in the

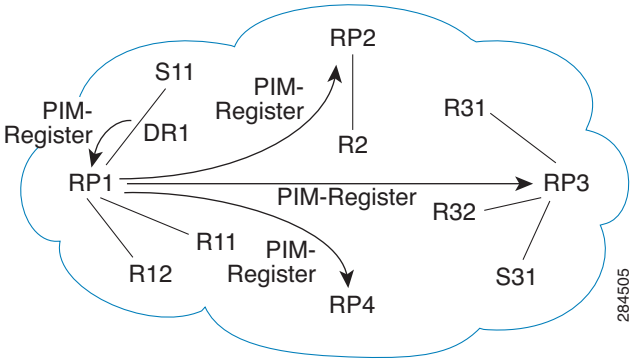
anycast RP set is configured with a loopback interface using the RP address. Each device in the anycast RP set also needs a separate physical IP address to be used for communication between the RPs.

The RP address, or a prefix that covers the RP address, is injected into the unicast routing system inside of the domain. Each device in the anycast RP set is configured with the addresses of all other devices in the anycast RP set, and this configuration must be consistent in all RPs in the set.

PIMv6 Anycast RP Normal Operation

The following illustration shows PIMv6 anycast RP normal operation and assumes the following:

- RP1, RP2, RP3, and RP4 are members in the same anycast RP group.
- S11 and S31 are sources that use RP1 and RP3, respectively, based on their unicast routing metric.
- R11, R12, R2, R31, and R32 are receivers. Based on their unicast routing metrics, R11 and R12 join to RP1, R2 joins to RP2 and R31, and R32 joins to RP3, respectively.

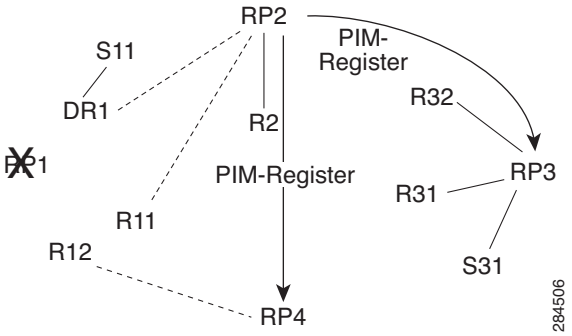


The following sequence of events occurs when S11 starts sending packets:

1. DR1 creates (S,G) states and sends a register to RP1. DR1 may also encapsulate the data packet in the register.
2. Upon receiving the register, RP1 performs normal PIM-SM RP functionality, and forwards the packets to R11 and R12.
3. RP1 also sends the register (which may encapsulate the data packets) to RP2, RP3, and RP4.
4. RP2, RP3, and RP4 do not further forward the register to each other.
5. RP2, RP3, and RP4 perform normal PIM-SM RP functionality, and if there is a data packet encapsulated, RP2 forwards the data packet to R2 and RP3 forwards the data packet to R31 and R32, respectively.
6. The previous five steps repeat for null registers sent by DR1.

PIMv6 Anycast RP Failover

The following illustration shows PIM anycast RP failover.



In failover, when RP1 is not reachable, the following occurs:

- Registers from DR1 will be routed transparently to RP2.
- R11 uses RP2 as the RP, and R12 uses RP4 as the RP.
- Registers from DR1 will be routed from RP2 to RP3 and RP4.

In this way, the loss of the RP (RP1 in this case) is transparent to DR1, R11, and R12, and the network can converge as soon as the IGP is converged.

How to Configure the PIMv6 Anycast RP Solution

- Configuring PIMv6 Anycast RP, page 421

Configuring PIMv6 Anycast RP

This task describes how to configure two PIMv6 anycast RP peers. Steps 3 through 11 show the configuration for RP1, and Steps 12 through 19 show the configuration for RP2.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ipv6 pim** [**vrf** *vrf-name*] **rp-address** *ipv6-address* [*group-address-list*] [**bidir**]
4. **interface** *type number*
5. **ipv6 address** {*ipv6-address*/*prefix-length* | *prefix-name sub-bits* /*prefix-length*}
6. **no shut**
7. **interface** *type number*
8. **ipv6 address** {*ipv6-address*/*prefix-length* | *prefix-name sub-bits* /*prefix-length*}
9. **no shut**
10. **exit**
11. **ipv6 pim anycast-RP** *rp-address peer-address*
12. **ipv6 pim** [**vrf** *vrf-name*] **rp-address** *ipv6-address* [*group-address-list*] [**bidir**]
13. **interface** *type number*
14. **ipv6 address** {*ipv6-address*/*prefix-length* | *prefix-name sub-bits* /*prefix-length*}
15. **no shut**
16. **interface** *type number*
17. **ipv6 address** {*ipv6-address*/*prefix-length* | *prefix-name sub-bits* /*prefix-length*}
18. **no shut**
19. **ipv6 pim anycast-RP** *rp-address peer-address*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**ipv6 pim** [**vrf** *vrf-name*] **rp-address** *ipv6-address* [*group-address-list*] [**bidir**] **Example:** `Device# ipv6 pim rp-address 2001:DB8::1:1 acl_sparse1`	Configures the address of a PIM RP for a particular group range.
Step 4	**interface** *type number* **Example:** `Device(config)# interface Loopback4`	Specifies an interface type and number, and places the device in interface configuration mode.
Step 5	**ipv6 address** {*ipv6-address*/*prefix-length* | *prefix-name sub-bits* / *prefix-length*} **Example:** `Device(config-if)# ipv6 address 2001:DB8::4:4/64`	Configures an IPv6 address based on an IPv6 general prefix and enable IPv6 processing on an interface.
Step 6	**no shut** **Example:** `Device(config-if)# no shut`	Enables an interface.
Step 7	**interface** *type number* **Example:** `Device(config-if)# interface Loopback5`	Specifies an interface type and number, and places the device in interface configuration mode.

	Command or Action	Purpose
Step 8	**ipv6 address** {*ipv6-address/prefix-length* | *prefix-name sub-bits* / *prefix-length*} **Example:** `Device(config-if)# ipv6 address 2001:DB8:0:ABCD::1/64`	Configures an IPv6 address based on an IPv6 general prefix and enable IPv6 processing on an interface.
Step 9	**no shut** **Example:** `Device(config-if)# no shut`	Enables an interface.
Step 10	**exit** **Example:** `Device(config-if)# exit`	Enter this command to exit interface configuration mode and enter global configuration mode.
Step 11	**ipv6 pim anycast-RP** *rp-address peer-address* **Example:** `Device(config)# ipv6 pim anycast-rp 2001:DB8:0:ABCD::1 2001:DB8::3:3`	Use this command to configure the address of the PIM RP for an anycast group range.
Step 12	**ipv6 pim** [**vrf** *vrf-name*] **rp-address** *ipv6-address* [*group-address-list*] [**bidir**] **Example:** `Device (config)# ipv6 pim rp-address 2001:DB8::1:1 acl_sparse1`	Configures the address of a PIM RP for a particular group range.
Step 13	**interface** *type number* **Example:** `Device(config)# interface Loopback4`	Specifies an interface type and number, and places the device in interface configuration mode.
Step 14	**ipv6 address** {*ipv6-address/prefix-length* | *prefix-name sub-bits* / *prefix-length*} **Example:** `Device(config-if)# ipv6 address 2001:DB8::3:3/64`	Configures an IPv6 address based on an IPv6 general prefix and enables IPv6 processing on an interface.

	Command or Action	Purpose
Step 15	**no shut** **Example:** `Device(config-if)# no shut`	Enables an interface.
Step 16	**interface** *type number* **Example:** `Device(config-if)# interface Loopback5`	Specifies an interface type and number, and places the device in interface configuration mode.
Step 17	**ipv6 address** {*ipv6-address/prefix-length* | *prefix-name sub-bits* / *prefix-length*} **Example:** `Device(config-if)# ipv6 address 2001:DB8:0:ABCD::1/64`	Configures an IPv6 address based on an IPv6 general prefix and enables IPv6 processing on an interface.
Step 18	**no shut** **Example:** `Device(config-if)# no shut`	Enables an interface
Step 19	**ipv6 pim anycast-RP** *rp-address peer-address* **Example:** `Device(config-if)# ipv6 pim anycast-rp 2001:DB8::1:1 2001:DB8::4:4`	Use this command to configure the address of the PIM RP for an anycast group range.

Configuration Examples for the PIMv6 Anycast RP Solution

Example: Configuring PIMv6 Anycast RP

```
RP1
Device1(config)# ipv6 pim rp-address 2001:DB8::1:1 acl_sparse1
Device1(config)# interface Loopback4
Device1(config-if)# ipv6 address 2001:DB8::4:4/64
Device1(config-if)# no shut

Device1(config)# interface Loopback5
Device1(config-if)# ipv6 address 2001:DB8:0:ABCD::1/64
Device1(config-if)# no shut
Device1(config-if)# exit
Device1(config)# ipv6 pim anycast-rp 2001:DB8:0:ABCD::1 2001:DB8::3:3

RP2 (Anycast RP Peer)
```

```
Device2(config)# ipv6 pim rp-address 2001:DB8::1:1 acl_sparse1
Device2(config)# interface Loopback4
Device2(config-if)# ipv6 address 2001:DB8::3:3/64
Device2(config-if)# no shut

Device2(config)# interface Loopback5
Device2(config-if)# ipv6 address 2001:DB8:0:ABCD::1/64
Device2(config-if)# no shut
Device2(config)# ipv6 pim anycast-rp 2001:DB8::1:1 2001:DB8::4:4
Device2 show ipv6 pim anycast-rp 2001:DB8::1:1

Anycast RP Peers For 2001:DB8::1:1   Last Register/Register-Stop received
  2001:DB8::3:3 00:00:00/00:00:00
  2001:DB8::4:4 00:00:00/00:00:00
```

Additional References

Related Documents

Related Topic	Document Title
Cisco IOS commands	*Cisco IOS Master Commands List, All Releases*
IPv6 commands	*Cisco IOS IPv6 Command Reference*
Cisco IOS IPv6 features	*Cisco IOS IPv6 Feature Mapping*

Standards and RFCs

Standard/RFC	Title
RFC 4610	*Anycast-RP Using Protocol Independent Multicast (PIM)*

Technical Assistance

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

Feature Information for PIMv6 Anycast RP Solution

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Table 18 Feature Information for the PIMv6: Anycast RP Solution

Feature Name	Releases	Feature Information
PIMv6: Anycast RP Solution	Cisco IOS XE Release 3.4S	The anycast RP solution in IPv6 PIM allows an IPv6 network to support anycast services for the PIM-SM RP. It allows anycast RP to be used inside a domain that runs PIM only. The following commands were introduced or modified: **ipv6 pim anycast-RP**, **show ipv6 pim anycast-RP**.

Implementing Multiprotocol BGP for IPv6

This module describes how to configure multiprotocol Border Gateway Protocol (BGP) for IPv6. BGP is an Exterior Gateway Protocol (EGP) used mainly to connect separate routing domains that contain independent routing policies (autonomous systems). Connecting to a service provider for access to the Internet is a common use for BGP. BGP can also be used within an autonomous system and this variation is referred to as internal BGP (iBGP). Multiprotocol BGP is an enhanced BGP that carries routing information for multiple network layer protocol address families, for example, IPv6 address family and for IP multicast routes. All BGP commands and routing policy capabilities can be used with multiprotocol BGP.

Finding Feature Information

Your software release may not support all the features documented in this module. For the latest caveats and feature information, see Bug Search Tool and the release notes for your platform and software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the feature information table at the end of this module.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Information About Implementing Multiprotocol BGP for IPv6

Multiprotocol BGP Extensions for IPv6

Multiprotocol BGP is the supported exterior gateway protocol (EGP) for IPv6. Multiprotocol BGP extensions for IPv6 supports many of the same features and functionality as IPv4 BGP. IPv6 enhancements to multiprotocol BGP include support for an IPv6 address family and network layer reachability

information (NLRI) and next hop (the next router in the path to the destination) attributes that use IPv6 addresses.

- IPv6 Multiprotocol BGP Peering Using a Link-Local Address, page 428

IPv6 Multiprotocol BGP Peering Using a Link-Local Address

The IPv6 multiprotocol BGP can be configured between two IPv6 devices (peers) using link-local addresses. For this function to work, the interface for the neighbor must be identified by using the **neighbor update-source** command, and a route map must be configured to set an IPv6 global next hop.

Multiprotocol BGP for the IPv6 Multicast Address Family

The multiprotocol BGP for the IPv6 multicast address family feature provides multicast BGP extensions for IPv6 and supports the same features and functionality as IPv4 BGP. IPv6 enhancements to multicast BGP include support for an IPv6 multicast address family and network layer reachability information (NLRI) and next hop (the next router in the path to the destination) attributes that use IPv6 addresses.

Multicast BGP is an enhanced BGP that allows the deployment of interdomain IPv6 multicast. Multiprotocol BGP carries routing information for multiple network layer protocol address families; for example, IPv6 address family and for IPv6 multicast routes. The IPv6 multicast address family contains routes used for RPF lookup by the IPv6 PIM protocol, and multicast BGP IPv6 provides for interdomain transport of the same. Users must use multiprotocol BGP for IPv6 multicast when using IPv6 multicast with BGP because the unicast BGP learned routes will not be used for IPv6 multicast.

Multicast BGP functionality is provided through a separate address family context. A subsequent address family identifier (SAFI) provides information about the type of the network layer reachability information that is carried in the attribute. Multiprotocol BGP unicast uses SAFI 1 messages, and multiprotocol BGP multicast uses SAFI 2 messages. SAFI 1 messages indicate that the routes are usable only for IP unicast, not IP multicast. Because of this functionality, BGP routes in the IPv6 unicast RIB must be ignored in the IPv6 multicast RPF lookup.

A separate BGP routing table is maintained to configure incongruent policies and topologies (for example, IPv6 unicast and multicast) by using IPv6 multicast RPF lookup. Multicast RPF lookup is very similar to the IP unicast route lookup.

No MRIB is associated with the IPv6 multicast BGP table. However, IPv6 multicast BGP operates on the unicast IPv6 RIB when needed. Multicast BGP does not insert or update routes into the IPv6 unicast RIB.

- Nonstop Forwarding and Graceful Restart for MP-BGP IPv6 Address Family, page 428

Nonstop Forwarding and Graceful Restart for MP-BGP IPv6 Address Family

The graceful restart capability is supported for IPv6 BGP unicast, multicast, and VPNv6 address families, enabling Cisco nonstop forwarding (NSF) functionality for BGP IPv6. The BGP graceful restart capability allows the BGP routing table to be recovered from peers without keeping the TCP state.

NSF continues forwarding packets while routing protocols converge, therefore avoiding a route flap on switchover. Forwarding is maintained by synchronizing the FIB between the active and standby RP. On switchover, forwarding is maintained using the FIB. The RIB is not kept synchronized; therefore, the RIB is empty on switchover. The RIB is repopulated by the routing protocols and subsequently informs FIB about RIB convergence by using the NSF_RIB_CONVERGED registry call. The FIB tables are updated from the RIB, removing any stale entries. The RIB starts a failsafe timer during RP switchover, in case the routing protocols fail to notify the RIB of convergence.

The Cisco BGP address family identifier (AFI) model is designed to be modular and scalable, and to support multiple AFI and subsequent address family identifier (SAFI) configurations.

How to Implement Multiprotocol BGP for IPv6

When configuring multiprotocol BGP extensions for IPv6, you must create the BGP routing process, configure peering relationships, and customize BGP for your particular network.

Note The following sections describe the configuration tasks for creating an IPv6 multiprotocol BGP routing process and associating peers, peer groups, and networks to the routing process. The following sections do not provide in-depth information on customizing multiprotocol BGP because the protocol functions the same in IPv6 as it does in IPv4. See the How to Implement Multiprotocol BGP for IPv6, page 429 section for further information on BGP and multiprotocol BGP configuration and command reference information.

Configuring an IPv6 BGP Routing Process and BGP Router ID

Perform this task to configure an IPv6 BGP routing process and an optional BGP router ID for a BGP-speaking device.

BGP uses a router ID to identify BGP-speaking peers. The BGP router ID is 32-bit value that is often represented by an IPv4 address. By default, the router ID is set to the IPv4 address of a loopback interface on the device. If no loopback interface is configured on the device, then the software chooses the highest IPv4 address configured to a physical interface on the device to represent the BGP router ID.

When configuring BGP on a device that is enabled only for IPv6 (the device does not have an IPv4 address), you must manually configure the BGP router ID for the device. The BGP router ID, which is represented as a 32-bit value using an IPv4 address syntax, must be unique to the BGP peers of the device.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **router bgp** *as-number*
4. **no bgp default ipv4-unicast**
5. **bgp router-id** *ip-address*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**router bgp** *as-number* **Example:** `Device(config)# router bgp 65000`	Configures a BGP routing process, and enters router configuration mode for the specified routing process.
Step 4	**no bgp default ipv4-unicast** **Example:** `Device(config-router)# no bgp default ipv4-unicast`	Disables the IPv4 unicast address family for the BGP routing process specified in the previous step. **Note** Routing information for the IPv4 unicast address family is advertised by default for each BGP routing session configured with the **neighbor remote-as** command unless you configure the **no bgp default ipv4-unicast** command before configuring the **neighbor remote-as** command.
Step 5	**bgp router-id** *ip-address* **Example:** `Device(config-router)# bgp router-id 192.168.99.70`	(Optional) Configures a fixed 32-bit router ID as the identifier of the local device running BGP. **Note** Configuring a router ID using the **bgp router-id** command resets all active BGP peering sessions.

Configuring IPv6 Multiprotocol BGP Between Two Peers

By default, neighbors that are defined using the **neighbor remote-as** command in router configuration mode exchange only IPv4 unicast address prefixes. To exchange other address prefix types, such as IPv6

prefixes, neighbors must also be activated using the **neighbor activate** command in address family configuration mode for the other prefix types, as shown for IPv6 prefixes.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **router bgp** *as-number*
4. **neighbor** {*ip-address* | *ipv6-address*[**%**] | *peer-group-name*} **remote-as** *autonomous-system-number* [**alternate-as** *autonomous-system-number* ...]
5. **address-family ipv6** [**unicast** | **multicast**]
6. **neighbor** {*ip-address* | *peer-group-name* | *ipv6-address* **%**} **activate**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**router bgp** *as-number* **Example:** `Device(config)# router bgp 65000`	Enters router configuration mode for the specified routing process.
Step 4	**neighbor** {*ip-address* | *ipv6-address*[**%**] | *peer-group-name*} **remote-as** *autonomous-system-number* [**alternate-as** *autonomous-system-number* ...] **Example:** `Device(config-router)# neighbor 2001:DB8:0:CC00::1 remote-as 64600`	Adds the IPv6 address of the neighbor in the specified autonomous system to the IPv6 multiprotocol BGP neighbor table of the local device.

	Command or Action	Purpose
Step 5	**address-family ipv6** [**unicast** | **multicast**] **Example:** `Device(config-router)# address-family ipv6`	Specifies the IPv6 address family and enters address family configuration mode. • The **unicast** keyword specifies the IPv6 unicast address family. By default, the device is placed in configuration mode for the IPv6 unicast address family if a keyword is not specified with the **address-family ipv6** command. • The **multicast** keyword specifies IPv6 multicast address prefixes.
Step 6	**neighbor** {*ip-address* | *peer-group-name* | *ipv6-address* **%**} **activate** **Example:** `Device(config-router-af)# neighbor 2001:DB8:0:CC00::1 activate`	Enables the neighbor to exchange prefixes for the IPv6 address family with the local device.

Configuring IPv6 Multiprotocol BGP Between Two Peers Using Link-Local Addresses

By default, neighbors that are defined using the **neighbor remote-as** command in router configuration mode exchange only IPv4 unicast address prefixes. To exchange other address prefix types, such as IPv6 prefixes, neighbors must also be activated using the **neighbor activate** command in address family configuration mode for the other prefix types, as shown for IPv6 prefixes.

By default, route maps that are applied in router configuration mode using the **neighbor route-map** command are applied to only IPv4 unicast address prefixes. Route maps for other address families must be applied in address family configuration mode using the **neighbor route-map** command, as shown for the IPv6 address family. The route maps are applied either as the inbound or outbound routing policy for neighbors under the specified address family. Configuring separate route maps under each address family type simplifies managing complicated or different policies for each address family.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **router bgp** *autonomous-system-number*
4. **neighbor** {*ip-address* | *ipv6-address* | *peer-group-name*} **remote-as** *as-number*
5. **neighbor** {*ip-address* | *ipv6-address* | *peer-group-name*} **update-source** *interface-type interface-number*
6. **address-family ipv6** [**vrf** *vrf-name*] [**unicast** | **multicast** | **vpnv6**
7. **neighbor** {*ip-address* | *peer-group-name* | *ipv6-address*} activate
8. **neighbor** {*ip-address* | *peer-group-name* | *ipv6-address*} **route-map** *map-name* {**in** | **out**
9. **exit**
10. Repeat Step 9.
11. **route-map** *map-tag* [**permit** | **deny**] [*sequence-number*]
12. **match ipv6 address** {**prefix-list** *prefix-list-name* | *access-list-name*
13. **set ipv6 next-hop** *ipv6-address* [*link-local-address*] [**peer-address**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**router bgp** *autonomous-system-number* **Example:** `Router(config)# router bgp 65000`	Enters router configuration mode for the specified routing process.
Step 4	**neighbor** {*ip-address* | *ipv6-address* | *peer-group-name*} **remote-as** *as-number* **Example:** `Router(config-router)# neighbor FE80::XXXX:BFF:FE0E:A471 remote-as 64600`	Adds the link-local IPv6 address of the neighbor in the specified remote autonomous system to the IPv6 multiprotocol BGP neighbor table of the local router. • The *ipv6-address*argument in the **neighbor remote-as**command must be a link-local IPv6 address in the form documented in RFC 2373 where the address is specified in hexadecimal using 16-bit values between colons.

	Command or Action	Purpose
Step 5	**neighbor** {*ip-address* | *ipv6-address* | *peer-group-name*} **update-source** *interface-type interface-number* **Example:** `Router(config-router)# neighbor FE80::XXXX:BFF:FE0E:A471 update-source gigabitethernet0/0/0`	Specifies the link-local address over which the peering is to occur. • If there are multiple connections to the neighbor and you do not specify the neighbor interface by using the *interface-type* and *interface-number*arguments in the **neighbor update-source** command, a TCP connection cannot be established with the neighbor using link-local addresses.
Step 6	**address-family ipv6** [**vrf** *vrf-name*] [**unicast** | **multicast** | **vpnv6** **Example:** `Router(config-router)# address-family ipv6`	Specifies the IPv6 address family, and enters address family configuration mode. • The **unicast** keyword specifies the IPv6 unicast address family. By default, the router is placed in configuration mode for the IPv6 unicast address family if the **unicast** keyword is not specified with the **address-family ipv6** command. • The **multicast** keyword specifies IPv6 multicast address prefixes.
Step 7	**neighbor** {*ip-address* | *peer-group-name* | *ipv6-address*} activate **Example:** `Router(config-router-af)# neighbor FE80::XXXX:BFF:FE0E:A471 activate`	Enables the neighbor to exchange prefixes for the IPv6 address family with the local router using the specified link-local addresses.
Step 8	**neighbor** {*ip-address* | *peer-group-name* | *ipv6-address*} **route-map** *map-name* {**in** | **out** **Example:** `Router(config-router-af)# neighbor FE80::XXXX:BFF:FE0E:A471 route-map nh6 out`	Applies a route map to incoming or outgoing routes.
Step 9	**exit** **Example:** `Router(config-router-af)# exit`	Exits address family configuration mode, and returns the router to router configuration mode.
Step 10	Repeat Step 9. **Example:** `Router(config-router)# exit`	Exits router configuration mode, and returns the router to global configuration mode.

Command or Action	Purpose
Step 11 **route-map** *map-tag* [**permit** | **deny**] [*sequence-number*] **Example:** `Router(config)# route-map nh6 permit 10`	Defines a route map and enters route-map configuration mode.
Step 12 **match ipv6 address** {**prefix-list** *prefix-list-name* | *access-list-name* **Example:** `Router(config-route-map)# match ipv6 address prefix-list cisco`	Distributes any routes that have a destination IPv6 network number address permitted by a prefix list, or performs policy routing on packets.
Step 13 **set ipv6 next-hop** *ipv6-address* [*link-local-address*] [**peer-address** **Example:** `Router(config-route-map)# set ipv6 next-hop 2001:DB8::1`	Overrides the next hop advertised to the peer for IPv6 packets that pass a match clause of a route map for policy routing. • The *ipv6-address* argument specifies the IPv6 global address of the next hop. It need not be an adjacent router. • The *link-local-address*argument specifies the IPv6 link-local address of the next hop. It must be an adjacent router. **Note** The route map sets the IPv6 next-hop addresses (global and link-local) in BGP updates. If the route map is not configured, the next-hop address in the BGP updates defaults to the unspecified IPv6 address (::), which is rejected by the peer. If you specify only the global IPv6 next-hop address (the *ipv6-address* argument) with the **set ipv6 next-hop**command after specifying the neighbor interface (the *interface-type* argument) with the **neighbor update-source** command in Configuring IPv6 Multiprotocol BGP Between Two Peers Using Link-Local Addresses, page 432, the link-local address of the interface specified with the *interface-type* argument is included as the next-hop in the BGP updates. Therefore, only one route map that sets the global IPv6 next-hop address in BGP updates is required for multiple BGP peers that use link-local addresses.

• Troubleshooting Tips, page 435

Troubleshooting Tips

If peering is not established by this task, it may be because of a missing route map **set ipv6 next-hop** command. Use the **debug bgp ipv6 update** command to display debugging information on the updates to help determine the state of the peering.

Configuring an IPv6 Multiprotocol BGP Peer Group

- By default, neighbors that are defined using the **neighbor remote-as** command in router configuration mode exchange only IPv4 unicast address prefixes. To exchange other address prefix types, such as IPv6 prefixes, neighbors must also be activated using the **neighbor activate** command in address family configuration mode for the other prefix types, as shown for IPv6 prefixes.
- By default, peer groups that are defined in router configuration mode using the **neighbor peer-group**command exchange only IPv4 unicast address prefixes. To exchange other address prefix types, such as IPv6 prefixes, you must activate peer groups using the **neighbor activate** command in address family configuration mode for the other prefix types, as shown for IPv6 prefixes.
- Members of a peer group automatically inherit the address prefix configuration of the peer group.
- IPv4 active neighbors cannot exist in the same peer group as active IPv6 neighbors. Create separate peer groups for IPv4 peers and IPv6 peers.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **router bgp** *as-number*
4. **neighbor** *peer-group-name* **peer-group**
5. **neighbor** {*ip-address* | *ipv6-address*[**%**] | *peer-group-name*} **remote-as** *autonomous-system-number* [**alternate-as** *autonomous-system-number* ...]
6. **address-family ipv6** [**vrf** *vrf-name*] [**unicast** | **multicast** | **vpnv6**
7. **neighbor** {*ip-address* | *peer-group-name* | *ipv6-address* **%**} **activate**
8. **neighbor** *ip-address* | *ipv6-address*} **send-label**
9. **neighbor** {*ip-address* | *ipv6-address*} **peer-group** *peer-group-name*
10. **exit**

DETAILED STEPS

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

	Command or Action	Purpose
Step 3	**router bgp** *as-number* **Example:** `Router(config)# router bgp 65000`	Enters router configuration mode for the specified BGP routing process.
Step 4	**neighbor** *peer-group-name* **peer-group** **Example:** `Router(config-router)# neighbor group1 peer-group`	Creates a multiprotocol BGP peer group.
Step 5	**neighbor** {*ip-address* | *ipv6-address*[**%**] | *peer-group-name*} **remote-as** *autonomous-system-number* [**alternate-as** *autonomous-system-number* ...] **Example:** `Router(config-router)# neighbor 2001:DB8:0:CC00::1 remote-as 64600`	Adds the IPv6 address of the neighbor in the specified autonomous system to the IPv6 multiprotocol BGP neighbor table of the local router.
Step 6	**address-family ipv6** [**vrf** *vrf-name*] [**unicast** | **multicast** | **vpnv6** **Example:** `Router(config-router)# address-family ipv6 unicast`	Specifies the IPv6 address family, and enters address family configuration mode. • The **unicast** keyword specifies the IPv6 unicast address family. By default, the router is placed in configuration mode for the IPv6 unicast address family if the **unicast** keyword is not specified with the **address-family ipv6** command. • The **multicast** keyword specifies IPv6 multicast address prefixes.
Step 7	**neighbor** {*ip-address* | *peer-group-name* | *ipv6-address* **%**} **activate** **Example:** `Router(config-router-af)# neighbor 2001:DB8:0:CC00::1 activate`	Enables the neighbor to exchange prefixes for the specified family type with the neighbor and the local router. • To avoid extra configuration steps for each neighbor, use the **neighbor activate** command with the *peer-group-name* argument as an alternative in this step.
Step 8	**neighbor** *ip-address* | *ipv6-address*} **send-label** **Example:** `Router(config-router-af)# neighbor 192.168.99.70 send-label`	Advertises the capability of the router to send MPLS labels with BGP routes. • In IPv6 address family configuration mode, this command enables binding and advertisement of aggregate labels when advertising IPv6 prefixes in BGP.

	Command or Action	Purpose
Step 9	**neighbor** {*ip-address* | *ipv6-address*} **peer-group** *peer-group-name* **Example:** `Router(config-router-af)# neighbor 2001:DB8:0:CC00::1 peer-group group1`	Assigns the IPv6 address of a BGP neighbor to a peer group.
Step 10	**exit** **Example:** `Router(config-router-af)# exit`	Exits address family configuration mode, and returns the router to router configuration mode. • Repeat this step to exit router configuration mode and return the router to global configuration mode.

Advertising IPv4 Routes Between IPv6 BGP Peers

If an IPv6 network is connecting two separate IPv4 networks, it is possible to use IPv6 to advertise the IPv4 routes. Configure the peering using the IPv6 addresses within the IPv4 address family. Set the next hop with a static route or with an inbound route map because the advertised next hop will usually be unreachable. Advertising IPv6 routes between two IPv4 peers is also possible using the same model.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **router bgp** *as-number*
4. **neighbor** *peer-group-name* **peer-group**
5. **neighbor** {*ip-address* | *ipv6-address*[**%**] | *peer-group-name*} **remote-as** *autonomous-system-number* [**alternate-as** *autonomous-system-number* ...]
6. **address-family ipv4** [**mdt** | **multicast** | **tunnel** | **unicast** [**vrf** *vrf-name*] | **vrf** *vrf-name*
7. **neighbor** *ipv6-address* **peer-group** *peer-group-name*
8. **neighbor** {*ip-address* | *peer-group-name* | *ipv6-address* [**%**]} **route-map** *map-name* {**in** | **out**}
9. **exit**
10. **exit**
11. **route-map** *map-tag* [**permit** | **deny**] [*sequence-number*]
12. **set ip next-hop** *ip-address* [... *ip-address*] [**peer-address**]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**router bgp** *as-number* **Example:** `Device(config)# router bgp 65000`	Enters router configuration mode for the specified routing process.
Step 4	**neighbor** *peer-group-name* **peer-group** **Example:** `Device(config-router)# neighbor 6peers peer-group`	Creates a multiprotocol BGP peer group.
Step 5	**neighbor** {*ip-address* | *ipv6-address*[**%**] | *peer-group-name*} **remote-as** *autonomous-system-number* [**alternate-as** *autonomous-system-number* ...] **Example:** `Device(config-router)# neighbor 6peers remote-as 65002`	Adds the IPv6 address of the neighbor in the specified autonomous system to the IPv6 multiprotocol BGP neighbor table of the local router.
Step 6	**address-family ipv4** [**mdt** | **multicast** | **tunnel** | **unicast** [**vrf** *vrf-name*] | **vrf** *vrf-name* **Example:** `Device(config-router)# address-family ipv4`	Enters address family configuration mode to configure a routing session using standard IPv4 address prefixes.

	Command or Action	Purpose
Step 7	**neighbor** *ipv6-address* **peer-group** *peer-group-name* **Example:** `Device(config-router-af)# neighbor 2001:DB8:1234::2 peer-group 6peers`	Assigns the IPv6 address of a BGP neighbor to a peer group.
Step 8	**neighbor** {*ip-address* | *peer-group-name* | *ipv6-address* [**%**]} **route-map** *map-name* {**in** | **out**} **Example:** `Device(config-router-af)# neighbor 6peers route-map rmap out`	Applies a route map to incoming or outgoing routes. • Changes to the route map will not take effect for existing peers until the peering is reset or a soft reset is performed. Using the **clear bgp ipv6** command with the **soft** and **in** keywords will perform a soft reset.
Step 9	**exit** **Example:** `Device(config-router-af)# exit`	Exits address family configuration mode, and returns the device to router configuration mode.
Step 10	**exit** **Example:** `Device(config-router)# exit`	Exits router configuration mode, and returns the device to global configuration mode.
Step 11	**route-map** *map-tag* [**permit** | **deny**] [*sequence-number*] **Example:** `Device(config)# route-map rmap permit 10`	Defines a route map and enters route-map configuration mode.
Step 12	**set ip next-hop** *ip-address* [*... ip-address*] [**peer-address**] **Example:** `Device(config-route-map)# set ip next-hop 10.21.8.10`	Overrides the next hop advertised to the peer for IPv4 packets.

Assigning BGP Administrative Distance for Multicast BGP Routes

Perform this task to specify an administrative distance for multicast BGP routes to be used in RPF lookups for comparison with unicast routes.

Caution Changing the administrative distance of BGP internal routes is not recommended. One problem that can occur is the accumulation of routing table inconsistencies, which can break routing.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **router bgp** *as-number*
4. **address-family ipv6** [**vrf** *vrf-name*] [**unicast** | **multicast** | **vpnv6**]
5. **distance bgp** *external-distance internal-distance local-distance*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**router bgp** *as-number* **Example:** `Router(config)# router bgp 65000`	Enters router configuration mode for the specified routing process.
Step 4	**address-family ipv6** [**vrf** *vrf-name*] [**unicast** | **multicast** | **vpnv6**] **Example:** `Router(config-router)# address-family ipv6`	Specifies the IPv6 address family, and enters address family configuration mode. • The **unicast** keyword specifies the IPv6 unicast address family. By default, the router is placed in configuration mode for the IPv6 unicast address family if the **unicast** keyword is not specified with the **address-family ipv6** command. • The **multicast** keyword specifies IPv6 multicast address prefixes.

	Command or Action	Purpose
Step 5	**distance bgp** *external-distance internal-distance local-distance* **Example:** `Router(config-router-af)# distance bgp 10 50 100`	Configures the administrative distance for BGP routes.

Generating IPv6 Multicast BGP Updates

Perform this task to generate IPv6 multicast BGP updates that correspond to unicast IPv6 updates received from a peer.

The MBGP translate-update feature generally is used in an MBGP-capable router that peers with a customer site that has only a BGP-capable router; the customer site has not or cannot upgrade its router to an MBGP-capable image. Because the customer site cannot originate MBGP advertisements, the router with which it peers will translate the BGP prefixes into MBGP prefixes, which are used for multicast-source Reverse Path Forwarding (RPF) lookup.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **router bgp** *as-number*
4. **address-family ipv6** [**vrf** *vrf-name*] [**unicast** | **multicast** | **vpnv6**
5. **neighbor** *ipv6-address* **translate-update ipv6 multicast** [**unicast**

DETAILED STEPS

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

Command or Action	Purpose
Step 3 **router bgp** *as-number* **Example:** `Router(config)# router bgp 65000`	Enters router configuration mode for the specified routing process.
Step 4 **address-family ipv6** [**vrf** *vrf-name*] [**unicast** | **multicast** | **vpnv6** **Example:** `Router(config-router)# address-family ipv6`	Specifies the IPv6 address family, and enters address family configuration mode. • The **unicast** keyword specifies the IPv6 unicast address family. By default, the router is placed in configuration mode for the IPv6 unicast address family if the **unicast** keyword is not specified with the **address-family ipv6** command. • The **multicast** keyword specifies IPv6 multicast address prefixes.
Step 5 **neighbor** *ipv6-address* **translate-update ipv6 multicast** [**unicast** **Example:** `Router(config-router-af)# neighbor 7000::2 translate-update ipv6 multicast`	Generates multiprotocol IPv6 BGP updates that correspond to unicast IPv6 updates received from a peer.

Configuring the IPv6 BGP Graceful Restart Capability

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **router bgp** *as-number*
4. **address-family ipv6** [**vrf** *vrf-name*] [**unicast** | **multicast** | **vpnv6**]
5. **bgp graceful-restart** [**restart-time** *seconds* | **stalepath-time** *seconds*] [**all**]

DETAILED STEPS

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

	Command or Action	Purpose
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**router bgp** *as-number* **Example:** `Device(config)# router bgp 65000`	Enters router configuration mode for the specified routing process.
Step 4	**address-family ipv6** [**vrf** *vrf-name*] [**unicast** | **multicast** | **vpnv6**] **Example:** `Device(config-router)# address-family ipv6`	Specifies the IPv6 address family.
Step 5	**bgp graceful-restart** [**restart-time** *seconds* | **stalepath-time** *seconds*] [**all**] **Example:** `Device(config-router-af)# bgp graceful-restart`	Enables the BGP graceful restart capability.

Resetting IPv6 BGP Sessions

SUMMARY STEPS

1. **enable**
2. **clear bgp ipv6** {**unicast** | **multicast**} {***** | *autonomous-system-number* | *ip-address* | *ipv6-address* | **peer-group** *peer-group-name*} [**soft**] [**in** | **out**]

DETAILED STEPS

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

	Command or Action	Purpose
Step 2	**clear bgp ipv6** {**unicast** | **multicast**} {***** | *autonomous-system-number* | *ip-address* | *ipv6-address* | **peer-group** *peer-group-name*} [**soft**] [**in** | **out**] **Example:** `Device# clear bgp ipv6 unicast peer-group marketing soft out`	Resets IPv6 BGP sessions.

Clearing External BGP Peers

SUMMARY STEPS

1. **enable**
2. **clear bgp ipv6** {**unicast** | **multicast**} **external** [**soft**] [**in** | **out**]
3. **clear bgp ipv6** {**unicast** | **multicast**} **peer-group** *name*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**clear bgp ipv6** {**unicast** | **multicast**} **external** [**soft**] [**in** | **out**] **Example:** `Device# clear bgp ipv6 unicast external soft in`	Clears external IPv6 BGP peers.
Step 3	**clear bgp ipv6** {**unicast** | **multicast**} **peer-group** *name* **Example:** `Device# clear bgp ipv6 unicast peer-group marketing`	Clears all members of an IPv6 BGP peer group.

Clearing IPv6 BGP Route Dampening Information

SUMMARY STEPS

1. **enable**
2. **clear bgp ipv6** {**unicast** | **multicast**} **dampening** [*ipv6-prefix prefix-length*]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**clear bgp ipv6** {**unicast** | **multicast**} **dampening** [*ipv6-prefix prefix-length*] **Example:** `Device# clear bgp ipv6 unicast dampening 2001:DB8::/64`	Clears IPv6 BGP route dampening information and unsuppresses the suppressed routes.

Clearing IPv6 BGP Flap Statistics

SUMMARY STEPS

1. **enable**
2. **clear bgp ipv6** {**unicast** | **multicast**} **flap-statistics** [*ipv6-prefix*/*prefix-length* | **regexp** *regexp* | **filter-list** *list*]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**clear bgp ipv6** {**unicast** | **multicast**} **flap-statistics** [*ipv6-prefix*/*prefix-length* | **regexp** *regexp* | **filter-list** *list*] **Example:** `Device# clear bgp ipv6 unicast flap-statistics filter-list 3`	Clears IPv6 BGP flap statistics.

Verifying IPv6 Multiprotocol BGP Configuration and Operation

SUMMARY STEPS

1. **show bgp ipv6 unicast** | **multicast**} [*ipv6-prefix*/*prefix-length*] [**longer-prefixes**] [**labels**]
2. **show bgp ipv6** {**unicast** | **multicast**} **summary**
3. **show bgp ipv6** {**unicast** | **multicast**} **dampening dampened-paths**
4. **enable**
5. **debug bgp ipv6** {**unicast** | **multicast**} **dampening**[**prefix-list** *prefix-list-name*]
6. **debug bgp ipv6 unicast** | **multicast**} **updates**[*ipv6-address*] [**prefix-list** *prefix-list-name*] [**in**| **out**]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**show bgp ipv6 unicast** | **multicast**} [*ipv6-prefix*/*prefix-length*] [**longer-prefixes**] [**labels**] **Example:** `Router> show bgp ipv6 unicast`	(Optional) Displays entries in the IPv6 BGP routing table.
Step 2	**show bgp ipv6** {**unicast** | **multicast**} **summary** **Example:** `Router> show bgp ipv6 unicast summary`	(Optional) Displays the status of all IPv6 BGP connections.
Step 3	**show bgp ipv6** {**unicast** | **multicast**} **dampening dampened-paths** **Example:** `Router> show bgp ipv6 unicast dampening dampened-paths`	(Optional) Displays IPv6 BGP dampened routes.
Step 4	**enable** **Example:** `Router> enable`	Enables higher privilege levels, such as privileged EXEC mode. • Enter your password if prompted.

	Command or Action	Purpose
Step 5	**debug bgp ipv6** {**unicast** | **multicast**} **dampening**[**prefix-list** *prefix-list-name*] **Example:** `Router# debug bgp ipv6 unicast dampening`	(Optional) Displays debugging messages for IPv6 BGP dampening packets. • If no prefix list is specified, debugging messages for all IPv6 BGP dampening packets are displayed.
Step 6	**debug bgp ipv6 unicast** | **multicast**} **updates**[*ipv6-address*] [**prefix-list** *prefix-list-name*] [**in**| **out**] **Example:** `Router# debug bgp ipv6 unicast updates`	(Optional) Displays debugging messages for IPv6 BGP update packets. • If an *ipv6-address* argument is specified, debugging messages for IPv6 BGP updates to the specified neighbor are displayed. • Use the **in** keyword to display debugging messages for inbound updates only. • Use the **out** keyword to display debugging messages for outbound updates only.

Configuration Examples for Multiprotocol BGP for IPv6

Example: Configuring a BGP Process, BGP Router ID, and IPv6 Multiprotocol BGP Peer

The following example enables IPv6 globally, configures a BGP process, and establishes a BGP router ID. Also, the IPv6 multiprotocol BGP peer 2001:DB8:0:CC00:: is configured and activated.

```
ipv6 unicast-routing
!
router bgp 65000
no bgp default ipv4-unicast
bgp router-id 192.168.99.70
neighbor 2001:DB8:0:CC00::1 remote-as 64600
address-family ipv6 unicast
  neighbor 2001:DB8:0:CC00::1 activate
```

Example Configuring an IPv6 Multiprotocol BGP Peer Using a Link-Local Address

The following example configures the IPv6 multiprotocol BGP peer FE80::XXXX:BFF:FE0E:A471 over Gigabit Ethernet interface 0/0/0 and sets the route map named nh6 to include the IPv6 next-hop global address of Gigabit Ethernet interface 0/0/0 in BGP updates. The IPv6 next-hop link-local address can be set

by the nh6 route map (not shown in the following example) or from the interface specified by the **neighbor update-source** command (as shown in the following example).

```
router bgp 65000
 neighbor FE80::XXXX:BFF:FE0E:A471 remote-as 64600
 neighbor FE80::XXXX:BFF:FE0E:A471 update-source gigabitethernet0/0/0
address-family ipv6
 neighbor FE80::XXXX:BFF:FE0E:A471 activate
 neighbor FE80::XXXX:BFF:FE0E:A471 route-map nh6 out
route-map nh6 permit 10
 match ipv6 address prefix-list cisco
 set ipv6 next-hop 2001:DB8:5y6::1
ipv6 prefix-list cisco permit 2001:DB8:2Fy2::/48 le 128
ipv6 prefix-list cisco deny ::/0
```

Note If you specify only the global IPv6 next-hop address (the *ipv6-address* argument) with the **set ipv6 next-hop** command after specifying the neighbor interface (the *interface-type* argument) with the **neighbor update-source** command, the link-local address of the interface specified with the *interface-type* argument is included as the next hop in the BGP updates. Therefore, only one route map that sets the global IPv6 next-hop address in BGP updates is required for multiple BGP peers that use link-local addresses.

Example: Configuring an IPv6 Multiprotocol BGP Peer Group

The following example configures the IPv6 multiprotocol BGP peer group named group1:

```
router bgp 65000
no bgp default ipv4-unicast
neighbor group1 peer-group
neighbor 2001:DB8:0:CC00::1 remote-as 64600
address-family ipv6 unicast
 neighbor group1 activate
 neighbor 2001:DB8:0:CC00::1 peer-group group1
```

Example: Advertising IPv4 Routes Between IPv6 Peers

The following example advertises IPv4 routes between IPv6 peers when the IPv6 network is connecting two separate IPv4 networks. Peering is configured using IPv6 addresses in the IPv4 address family configuration mode. The inbound route map named rmap sets the next hop because the advertised next hop is likely to be unreachable.

```
router bgp 65000
!
 neighbor 6peers peer-group
 neighbor 2001:DB8:1234::2 remote-as 65002
 address-family ipv4
 neighbor 6peers activate
 neighbor 6peers soft-reconfiguration inbound
 neighbor 2001:DB8:1234::2 peer-group 6peers
 neighbor 2001:DB8:1234::2 route-map rmap in
!
route-map rmap permit 10
 set ip next-hop 10.21.8.10
```

Where to Go Next

If you want to implement more IPv6 routing protocols, refer to the Implementing RIP for IPv6 or the Implementing IS-IS for IPv6 module.

- Additional References, page 450

Additional References

Related Documents

Related Topic	Document Title
BGP and multiprotocol BGP commands: complete command syntax, command mode, defaults, usage guidelines, and examples	" BGP Commands ," *Cisco IOS IP Routing Protocols Command Reference*
Cisco Nonstop Forwarding	" Cisco Nonstop Forwarding ," *Cisco IOS XE High Availability Configuration Guide*
IPv6 supported feature list	"Start Here: Cisco IOS XE Software Release Specifics for IPv6 Features ," *Cisco IOS XE IPv6 Configuration Guide*
IPv6 commands: complete command syntax, command mode, defaults, usage guidelines, and examples	*Cisco IOS IPv6 Command Reference*
Cisco IOS master command list, all releases	Cisco IOS Master Command List, All Releases

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
None	To locate and download MIBs for selected platforms, Cisco software releases, and feature sets, use Cisco MIB Locator found at the following URL: http://www.cisco.com/go/mibs

RFCs

RFCs	Title
RFC 2545	*Use of BGP-4 Multiprotocol Extensions for IPv6 Inter-Domain Routing*
RFC 2858	*Multiprotocol Extensions for BGP-4*
RFC 4007	*IPv6 Scoped Address Architecture*
RFC 4364	BGP MPLS/IP Virtual Private Networks (VPNs)
RFC 4382	MPLS/BGP Layer 3 Virtual Private Network (VPN) Management Information Base
RFC 4659	BGP-MPLS IP Virtual Private Network (VPN) Extension for IPv6 VPN
RFC 4724	*Graceful Restart Mechanism for BGP*

Technical Assistance

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

Feature Information for Implementing Multiprotocol BGPfor IPv6

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Table 19 Feature Information for Implementing Multiprotocol BGP for IPv6

Feature Name	Releases	Feature Information
IPv6--NSF and Graceful Restart for MP-BGP IPv6 Address Family	Cisco IOS XE Release 3.1S	IPv6 BGP supports Cisco Nonstop Forwarding and graceful restart.
IPv6 Multicast Address Family Support for Multiprotocol BGP	Cisco IOS XE Release 2.1	The multiprotocol BGP for the IPv6 multicast address family feature provides multicast BGP extensions for IPv6 and supports the same features and functionality as IPv4 BGP.
IPv6 Routing--Multiprotocol BGP Extensions for IPv6	Cisco IOS XE Release 2.1	Multiprotocol BGP extensions for IPv6 supports the same features and functionality as IPv4 BGP.
IPv6 Routing--Multiprotocol BGP Link-Local Address Peering	Cisco IOS XE Release 2.1	IPv6 supports multiprotocol BGP link-local address peering.

Implementing NAT-PT for IPv6

Network Address Translation--Protocol Translation (NAT-PT) is an IPv6 to IPv4 translation mechanism, as defined in RFC 2765 and RFC 2766, allowing IPv6-only devices to communicate with IPv4-only devices and vice versa.

Finding Feature Information

Your software release may not support all the features documented in this module. For the latest caveats and feature information, see Bug Search Tool and the release notes for your platform and software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the feature information table at the end of this module.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Prerequisites for Implementing NAT-PT for IPv6

Before implementing NAT-PT, you must configure IPv4 and IPv6 on the router interfaces that need to communicate between IPv4-only and IPv6-only networks.

Restrictions for Implementing NAT-PT for IPv6

- NAT-PT is not supported in Cisco Express Forwarding.
- NAT-PT provides limited Application Layer Gateway (ALG) support--ALG support for Internet Control Message Protocol (ICMP), File Transfer Protocol (FTP), and Domain Naming System (DNS).
- NAT-PT has the same restrictions that apply to IPv4 NAT where NAT-PT does not provide end-to-end security and the NAT-PT router can be a single point of failure in the network.

- Users must decide whether to use Static NAT-PT operation, Dynamic NAT-PT operation, Port Address Translation (PAT), or IPv4-mapped operation. Deciding which operation to use determines how a user will configure and operate NAT-PT.
- Bridge-Group Virtual interfaces (BVIs) in IPv6 are not supported with NAT-PT and wireless interfaces Dot11Radio.

Information About Implementing NAT-PT for IPv6

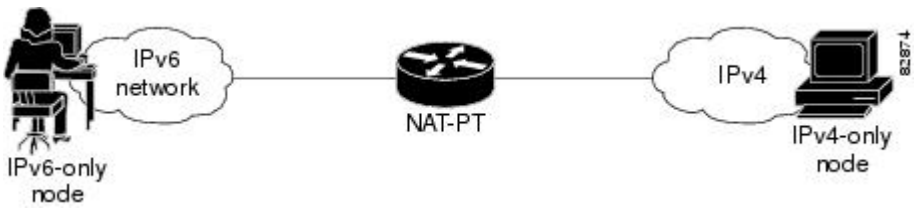
Users can configure NAT-PT using one of the following operations--static NAT-PT, dynamic NAT-PT, Port Address Translation (PAT), or IPv4-mapped operation--which are described in the following sections:

NAT-PT Overview

NAT-PT for Cisco software was designed using RFC 2766 and RFC 2765 as a migration tool to help customers transition their IPv4 networks to IPv6 networks. Using a protocol translator between IPv6 and IPv4 allows direct communication between hosts speaking a different network protocol. Users can use either static definitions or IPv4-mapped definitions for NAT-PT operation.

The figure below shows that NAT-PT runs on a router between an IPv6 network and an IPv4 network to connect an IPv6-only node with an IPv4-only node.

Figure 39 NAT-PT Basic Operation



Although IPv6 solves addressing issues for customers, a long transition period is likely before customers move to an exclusive IPv6 network environment. During the transition period, any new IPv6-only networks will need to continue to communicate with existing IPv4 networks. NAT-PT is designed to be deployed to allow direct communication between IPv6-only networks and IPv4-only networks. For a service provider customer, an example could be an IPv6-only client trying to access an IPv4-only web server. Enterprise customers will also migrate to IPv6 in stages, and many of their IPv4-only networks will be operational for several years. Dual-stack networks may have some IPv6-only hosts configured to take advantage of the IPv6 autoconfiguration, global addressing, and simpler management, and these hosts can use NAT-PT to communicate with existing IPv4-only networks in the same organization.

One of the benefits of NAT-PT is that no changes are required to existing hosts, because all the NAT-PT configurations are performed at the NAT-PT router. Customers with existing stable IPv4 networks can introduce an IPv6 network and use NAT-PT to allow communication without disrupting the existing network. To further illustrate the seamless transition, File Transfer Protocol (FTP) can be used between IPv4 and IPv6 networks, just as within an IPv4 network. Packet fragmentation is enabled by default when

IPv6 is configured, allowing IPv6 and IPv4 networks to resolve fragmentation problems between the networks. Without the ability to resolve fragmentation, connectivity could become intermittent when fragmented packets might be dropped or improperly interpreted.

Cisco has developed other transition techniques including dual stack, IPv6 over MPLS, and tunneling. NAT-PT should not be used when other native communication techniques exist. If a host is configured as a dual-stack host with both IPv4 and IPv6, we do not recommend using NAT-PT to communicate between the dual-stack host and an IPv6-only or IPv4-only host. NAT-PT is not recommended for a scenario in which an IPv6-only network is trying to communicate to another IPv6-only network via an IPv4 backbone or vice versa, because NAT-PT would require a double translation to be performed. In this scenario, tunneling techniques are recommended.

The following sections describe the operations that may be used to configure NAT-PT. Users have the option to use one of the following operations for NAT-PT operation, but not all four.

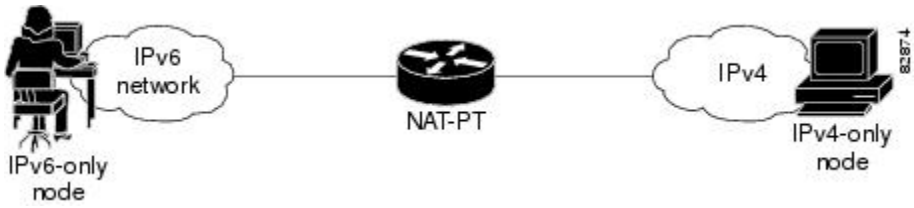
Static NAT-PT Operation

Static NAT-PT uses static translation rules to map one IPv6 address to one IPv4 address. IPv6 network nodes communicate with IPv4 network nodes using an IPv6 mapping of the IPv4 address configured on the NAT-PT router.

The figure below shows how the IPv6-only node named A can communicate with the IPv4-only node named C using NAT-PT. The NAT-PT device is configured to map the source IPv6 address for node A of 2001:DB8:bbbb:1::1 to the IPv4 address 192.168.99.2. NAT-PT is also configured to map the source address of IPv4 node C, 192.168.30.1 to 2001:DB8::a. When packets with a source IPv6 address of node A are received at the NAT-PT router, they are translated to have a destination address to match node C in the IPv4-only network. NAT-PT can also be configured to match a source IPv4 address and translate the packet to an IPv6 destination address to allow an IPv4-only host communicate with an IPv6-only host.

If you have multiple IPv6-only or IPv4-only hosts that need to communicate, you may need to configure many static NAT-PT mappings. Static NAT-PT is useful when applications or servers require access to a stable IPv4 address, such as accessing an external IPv4 DNS server.

Figure 40 Static NAT-PT Operation



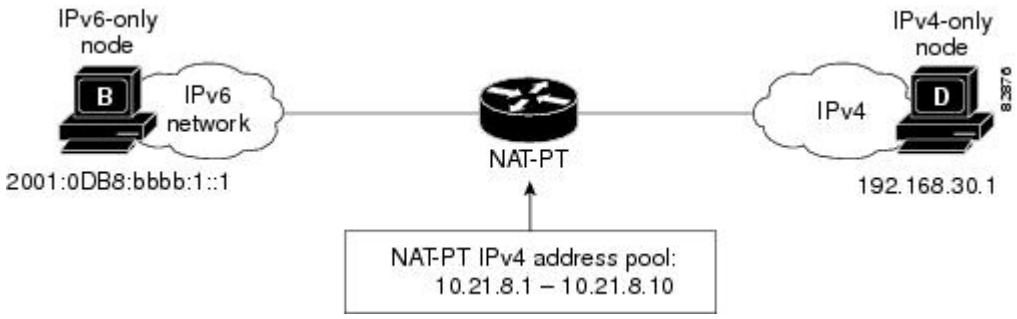
Dynamic NAT-PT Operation

Dynamic NAT-PT allows multiple NAT-PT mappings by allocating addresses from a pool. NAT-PT is configured with a pool of IPv6 and/or IPv4 addresses. At the start of a NAT-PT session a temporary address is dynamically allocated from the pool. The number of addresses available in the address pool determines the maximum number of concurrent sessions. The NAT-PT device records each mapping between addresses in a dynamic state table.

The figure below shows how dynamic NAT-PT operates. The IPv6-only node B can communicate with the IPv4-only node D using dynamic NAT-PT. The NAT-PT device is configured with an IPv6 access list, prefix list, or route map to determine which packets are to be translated by NAT-PT. A pool of IPv4 addresses--10.21.8.1 to 10.21.8.10 in the figure -- is also configured. When an IPv6 packet to be translated

is identified, NAT-PT uses the configured mapping rules and assigns a temporary IPv4 address from the configured pool of IPv4 addresses.

Figure 41 Dynamic NAT-PT Operation



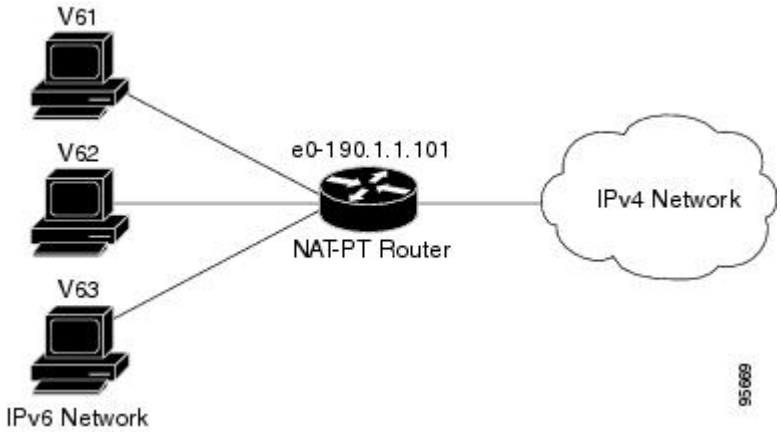
Dynamic NAT-PT translation operation requires at least one static mapping for the IPv4 DNS server.

After the IPv6 to IPv4 connection is established, the reply packets going from IPv4 to IPv6 take advantage of the previously established dynamic mapping to translate back from IPv4 to IPv6. If the connection is initiated by an IPv4-only host, then the explanation is reversed.

Port Address Translation or Overload

Port Address Translation (PA T), also known as Overload, allows a single IPv4 address to be used among multiple sessions by multiplexing on the port number to associate several IPv6 users with a single IPv4 address. PAT can be accomplished through a specific interface or through a pool of addresses. The figure below shows multiple IPv6 addresses from the IPv6 network linked to a single IPv4 interface into the IPv4 network.

Figure 42 Port Address Translation



IPv4-Mapped Operation

Customers can also send traffic from their IPv6 network to an IPv4 network without configuring IPv6 destination address mapping. A packet arriving at an interface is checked to discover if it has a NAT-PT prefix that was configured with the **ipv6 nat prefix v4-mapped** command. If the prefix matches, then an

access-list check is performed to discover if the source address matches the access list or prefix list. If the prefix does not match, the packet is dropped.

If the prefix matches, source address translation is performed. If a rule has been configured for the source address translation, the last 32 bits of the destination IPv6 address is used as the IPv4 destination and a flow entry is created.

With an IPv4-mapping configuration on the router, when the DNS ALG IPv4 address is converted to an IPv6 address, the IPv6 address is processed and the DNS packets from IPv4 network get their ALGs translated into the IPv6 network.

How to Implement NAT-PT for IPv6

Configuring Basic IPv6 to IPv4 Connectivity for NAT-PT for IPv6

Perform this task to configure basic IPv6 to IPv4 connectivity for NAT-PT, which consists of configuring the NAT-PT prefix globally, and enable NAT-PT on an interface. For NAT-PT to be operational, NAT-PT must be enabled on both the incoming and outgoing interfaces.

An IPv6 prefix with a prefix length of 96 must be specified for NAT-PT to use. The IPv6 prefix can be a unique local unicast prefix, a subnet of your allocated IPv6 prefix, or even an extra prefix obtained from your Internet service provider (ISP). The NAT-PT prefix is used to match a destination address of an IPv6 packet. If the match is successful, NAT-PT will use the configured address mapping rules to translate the IPv6 packet to an IPv4 packet. The NAT-PT prefix can be configured globally or with different IPv6 prefixes on individual interfaces. Using a different NAT-PT prefix on several interfaces allows the NAT-PT router to support an IPv6 network with multiple exit points to IPv4 networks.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ipv6 nat prefix** *ipv6-prefix* / *prefix-length*
4. **interface** *type number*
5. **ipv6 address** *ipv6-address* {/*prefix-length* | **link-local**}
6. **ipv6 nat**
7. **exit**
8. **interface** *type number*
9. **ip address** *ip-address mask* [**secondary**]
10. **ipv6 nat**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**ipv6 nat prefix** *ipv6-prefix* / *prefix-length* **Example:** `Router# ipv6 nat prefix 2001:DB8::/96`	Assigns an IPv6 prefix as a global NAT-PT prefix. • Matching destination prefixes in IPv6 packets are translated by NAT-PT. • The only prefix length supported is 96.
Step 4	**interface** *type number* **Example:** `Router(config)# interface gigabitethernet 3/1/1`	Specifies an interface type and number, and places the router in interface configuration mode.
Step 5	**ipv6 address** *ipv6-address* {/*prefix-length* | **link-local**} **Example:** `Router(config-if)# ipv6 address 2001:DB8:yyyy:1::9/64`	Specifies an IPv6 address assigned to the interface and enables IPv6 processing on the interface.
Step 6	**ipv6 nat** **Example:** `Router(config-if)# ipv6 nat`	Enables NAT-PT on the interface.
Step 7	**exit** **Example:** `Router(config-if)# exit`	Exits interface configuration mode, and returns the router to global configuration mode.

	Command or Action	Purpose
Step 8	**interface** *type number* **Example:** `Router(config)# interface gigabitethernet 3/3/1`	Specifies an interface type and number, and places the router in interface configuration mode.
Step 9	**ip address** *ip-address mask* [**secondary**] **Example:** `Router(config-if)# ip address 192.168.30.9 255.255.255.0`	Specifies an IP address and mask assigned to the interface and enables IP processing on the interface.
Step 10	**ipv6 nat** **Example:** `Router(config-if)# ipv6 nat`	Enables NAT-PT on the interface.

Configuring IPv4-Mapped NAT-PT

Perform this task to enable customers to send traffic from their IPv6 network to an IPv4 network without configuring IPv6 destination address mapping. This task shows the **ipv6 nat prefix v4-mapped** command configured on a specified interface, but the command could alternatively be configured globally:

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type number*
4. **ipv6 nat prefix** *ipv6-prefix* **v4-mapped** {*access-list-name* | *ipv6-prefix*}

DETAILED STEPS

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

	Command or Action	Purpose
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**interface** *type number* **Example:** `Router(config)# interface gigabitethernet 3/1/1`	Specifies an interface type and number, and places the router in interface configuration mode.
Step 4	**ipv6 nat prefix** *ipv6-prefix* **v4-mapped** {*access-list-name* | *ipv6-prefix*} **Example:** `Router(config-if)# ipv6 nat prefix 2001::/96 v4-mapped v4mapacl`	Enables customers to send traffic from their IPv6 network to an IPv4 network without configuring IPv6 destination address mapping.

Configuring Mappings for IPv6 Hosts Accessing IPv4 Hosts

Perform this task to configure static or dynamic IPv6 to IPv4 address mappings. The dynamic address mappings include assigning a pool of IPv4 addresses and using an access list, prefix list, or route map to define which packets are to be translated.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. Do one of the following:
   - **ipv6 nat v6v4 source** *ipv6-address ipv4-address*
   -
   - **ipv6 nat v6v4 source** {**list** *access-list-name* | **route-map** *map-name*} **pool** *name*
4. **ipv6 nat v6v4 pool** *name start-ipv4 end-ipv4* **prefix-length** *prefix-length*
5. **ipv6 nat translation** [**max-entries** *number*] {**timeout** | **udp-timeout** | **dns-timeout**| **tcp-timeout**| **finrst-timeout**| **icmp-timeout**} {*seconds*| **never**}
6. **ipv6 access-list** *access-list-name*
7. **permit** *protocol* {*source-ipv6-prefix / prefix-length*| **any**| **host** *source-ipv6-address*}[*operator* [*port-number*]] {*destination-ipv6-prefix / prefix-length*| **any**| **host** *destination-ipv6-address*}
8. **exit**
9. **show ipv6 nat translations** [**icmp**| **tcp**| **udp**] [**verbose**]
10. **show ipv6 nat statistics**

DETAILED STEPS

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

Step 3	Do one of the following:
	• **ipv6 nat v6v4 source** *ipv6-address ipv4-address*
	•
	• **ipv6 nat v6v4 source** {**list** *access-list-name* | **route-map** *map-name*} **pool** *name*
	
	Example:
	`Router(config)# ipv6 nat v6v4 source 2001:DB8:yyyy:1::1 10.21.8.10`
	
	Example:
	
	Example:
	`Router(config)# ipv6 nat v6v4 source list pt-list1 pool v4pool`
		Enables a static IPv6 to IPv4 address mapping using NAT-PT.
		or
		Enables a dynamic IPv6 to IPv4 address mapping using NAT-PT.
		• Use the **list** or **route-map** keyword to specify a prefix list, access list, or a route map to define which packets are translated.
		• Use the **pool** keyword to specify the name of a pool of addresses, created by the **ipv6 nat v6v4 pool**command, to be used in dynamic NAT-PT address mapping.

Step 4	**ipv6 nat v6v4 pool** *name start-ipv4 end-ipv4* **prefix-length** *prefix-length* **Example:** `Router(config)# ipv6 nat v6v4 pool v4pool 10.21.8.1 10.21.8.10 prefix-length 24`	Specifies a pool of IPv4 addresses to be used by NAT-PT for dynamic address mapping.

	Command or Action	Purpose
Step 5	**ipv6 nat translation** [**max-entries** *number*] {**timeout** | **udp-timeout** | **dns-timeout**| **tcp-timeout**| **finrst-timeout**| **icmp-timeout**} {*seconds*| **never**} **Example:** `Router(config)# ipv6 nat translation udp-timeout 600`	(Optional) Specifies the time after which NAT-PT translations time out.
Step 6	**ipv6 access-list** *access-list-name* **Example:** `Router(config)# ipv6 access-list pt-list1`	(Optional) Defines an IPv6 access list and enters IPv6 access list configuration mode. The router prompt changes to Router(config-ipv6-acl)#. • The *access-list name* argument specifies the name of the IPv6 access control list (ACL). IPv6 ACL names cannot contain a space or quotation mark, or begin with a numeral.
Step 7	**permit** *protocol* {*source-ipv6-prefix* / *prefix-length*| **any**| **host** *source-ipv6-address*}[*operator* [*port-number*]] {*destination-ipv6-prefix* / *prefix-length*| **any**| **host** *destination-ipv6-address*} **Example:** `Router(config-ipv6-acl)# permit ipv6 2001:DB8:bbbb:1::/64 any`	(Optional) Specifies permit conditions for an IPv6 ACL. • The *protocol* argument specifies the name or number of an Internet protocol. It can be one of the keywords **ahp**, **esp**, **icmp**, **ipv6**, **pcp**, **sctp**, **tcp**, or **udp**, or an integer in the range from 0 to 255 representing an IPv6 protocol number. • The *source-ipv6-prefix* / *prefix-length* and *destination-ipv6-prefix* / *prefix-length* arguments specify the source and destination IPv6 network or class of networks about which to set permit conditions. These arguments must be in the form documented in RFC 2373 where the address is specified in hexadecimal using 16-bit values between colons. • The **any** keyword is an abbreviation for the IPv6 prefix ::/0. • The **host** *source-ipv6-address* keyword and argument combination specifies the source IPv6 host address about which to set permit conditions. The *source-ipv6-address*argument must be in the form documented in RFC 2373 where the address is specified in hexadecimal using 16-bit values between colons.
Step 8	**exit** **Example:** `Router(config-if)# exit`	Exits access list configuration mode, and returns the router to global configuration mode. Enter the **exit** command twice to return to privileged EXEC mode.

	Command or Action	Purpose
Step 9	**show ipv6 nat translations** [**icmp**| **tcp**| **udp**] [**verbose**] **Example:** `Router# show ipv6 nat translations verbose`	(Optional) Displays active NAT-PT translations. • Use the optional **icmp**, **tcp**, and **udp** keywords to display detailed information about the NAT-PT translation events for the specified protocol. • Use the optional **verbose** keyword to display more detailed information about the active translations.
Step 10	**show ipv6 nat statistics** **Example:** `Router# show ipv6 nat statistics`	(Optional) Displays NAT-PT statistics.

Configuring Mappings for IPv4 Hosts Accessing IPv6 Hosts

Perform this optional task to configure static or dynamic IPv4 to IPv6 address mappings. The dynamic address mappings include assigning a pool of IPv6 addresses and using an access list, prefix list, or route map to define which packets are to be translated.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. Do one of the following:
   - **ipv6 nat v4v6 source** *ipv6-address ipv4-address*
   -
   - **ipv6 nat v4v6 source list** {*access-list-number* | *name*} **pool** *name*
4. **ipv6 nat v4v6 pool** *name start-ipv6 end-ipv6* **prefix-length** *prefix-length*
5. **access-list** {*access-list-name*| *number*}{**deny**| **permit**} [*source source-wildcard*] [**log**]

DETAILED STEPS

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

	Command or Action	Purpose
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.

Step 3	Do one of the following:
	• **ipv6 nat v4v6 source** *ipv6-address ipv4-address*
	•
	• **ipv6 nat v4v6 source list** {*access-list-number* | *name*} **pool** *name*
	
	Example:
	`Router(config)# ipv6 nat v4v6 source 10.21.8.11 2001:DB8:yyyy::2`
	
	Example:
	
	Example:
	`Router(config)# ipv6 nat v4v6 source list 1 pool v6pool`
		Enables a static IPv4 to IPv6 address mapping using NAT-PT.
		or
		Enables a dynamic IPv4 to IPv6 address mapping using NAT-PT.
		• Use the **list** keyword to specify an access list to define which packets are translated.
		• Use the **pool** keyword to specify the name of a pool of addresses, created by the **ipv6 nat v4v6 pool**command, to be used in dynamic NAT-PT address mapping.

Step 4	**ipv6 nat v4v6 pool** *name start-ipv6 end-ipv6* **prefix-length** *prefix-length* **Example:** `Router(config)# ipv6 nat v4v6 pool v6pool 2001:DB8:yyyy::1 2001:DB8:yyyy::2 prefix-length 128`	Specifies a pool of IPv6 addresses to be used by NAT-PT for dynamic address mapping.
Step 5	**access-list** {*access-list-name*| *number*}{**deny**| **permit**} [*source source-wildcard*] [**log**] **Example:** `Router(config)# access-list 1 permit 192.168.30.0 0.0.0.255`	Specifies an entry in a standard IPv4 access list.

Configuring PAT for IPv6 to IPv4 Address Mappings

Perform this task to configure PAT for IPv6 to IPv4 address mappings. Multiple IPv6 addresses are mapped to a single IPv4 address or to a pool of IPv4 addresses and using an access list, prefix list, or route map to define which packets are to be translated.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. Do one of the following:
   - **ipv6 nat v6v4 source** {**list** *access-list-name* | **route-map** *map-name*} **pool** *name overload*
   -
   - **ipv6 nat v6v4 source** {**list** *access-list-name* | **route-map** *map-name*} **interface** *interface name* **overload**
4. **ipv6 nat v6v4 pool** *name start-ipv4 end-ipv4* **prefix-length** *prefix-length*
5. **ipv6 nat translation** [**max-entries** *number*] {**timeout** | **udp-timeout** | **dns-timeout**| **tcp-timeout**| **finrst-timeout**| **icmp-timeout**} {*seconds*| **never**}
6. **ipv6 access-list** *access-list-name*
7. **permit** *protocol* {*source-ipv6-prefix / prefix-length*| **any**| **host** *source-ipv6-address*}[*operator* [*port-number*]] {*destination-ipv6-prefix / prefix-length*| **any**| **host** *destination-ipv6-address*}

DETAILED STEPS

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

Command or Action	Purpose

Step 3 Do one of the following:
• **ipv6 nat v6v4 source** {**list** *access-list-name* | **route-map** *map-name*} **pool** *name overload*
•
• **ipv6 nat v6v4 source** {**list** *access-list-name* | **route-map** *map-name*} **interface** *interface name* **overload**
Example:
`Router(config)# ipv6 nat v6v4 source 2001:DB8:yyyy:1::1 10.21.8.10`
Example:
Example:
`Router(config)# ipv6 nat v6v4 source list pt-list1 pool v4pool overload`
	Enables a dynamic IPv6 to IPv4 address overload mapping using a pool address.
	or
	Enables a dynamic IPv6 to IPv4 address overload mapping using an interface address.
	• Use the **list** or **route-map** keyword to specify a prefix list, access list, or a route map to define which packets are translated.
	• Use the **pool** keyword to specify the name of a pool of addresses, created by the **ipv6 nat v6v4 pool**command, to be used in dynamic NAT-PT address mapping.
	• Use the interface keyword to specify the interface address to be used for overload.

Step 4 **ipv6 nat v6v4 pool** *name start-ipv4 end-ipv4* **prefix-length** *prefix-length* **Example:** `Router(config)# ipv6 nat v6v4 pool v4pool 10.21.8.1 10.21.8.10 prefix-length 24`	Specifies a pool of IPv4 addresses to be used by NAT-PT for dynamic address mapping.
Step 5 **ipv6 nat translation** [**max-entries** *number*] {**timeout** | **udp-timeout** | **dns-timeout**| **tcp-timeout**| **finrst-timeout**| **icmp-timeout**} {*seconds*| **never**} **Example:** `Router(config)# ipv6 nat translation udp-timeout 600`	(Optional) Specifies the time after which NAT-PT translations time out.

	Command or Action	Purpose
Step 6	**ipv6 access-list** *access-list-name* **Example:** `Router(config)# ipv6 access-list pt-list1`	(Optional) Defines an IPv6 access list and enters IPv6 access list configuration mode. The router prompt changes to Router(config-ipv6-acl)#. • The *access-list name* argument specifies the name of the IPv6 access control list (ACL). IPv6 ACL names cannot contain a space or quotation mark, or begin with a numeral.
Step 7	**permit** *protocol* {*source-ipv6-prefix* / *prefix-length*| **any**| **host** *source-ipv6-address*}[*operator* [*port-number*]] {*destination-ipv6-prefix* / *prefix-length*| **any**| **host** *destination-ipv6-address*} **Example:** `Router(config-ipv6-acl)# permit ipv6 2001:DB8:bbbb:1::/64 any`	(Optional) Specifies permit conditions for an IPv6 ACL. • The *protocol* argument specifies the name or number of an Internet protocol. It can be one of the keywords **ahp**, **esp**, **icmp**, **ipv6**, **pcp**, **sctp**, **tcp**, or **udp**, or an integer in the range from 0 to 255 representing an IPv6 protocol number. • The *source-ipv6-prefix* / *prefix-length* and *destination-ipv6-prefix* / *prefix-length* arguments specify the source and destination IPv6 network or class of networks about which to set permit conditions. These arguments must be in the form documented in RFC 2373 where the address is specified in hexadecimal using 16-bit values between colons. • The **any** keyword is an abbreviation for the IPv6 prefix ::/0. • The **host** *source-ipv6-address* keyword and argument combination specifies the source IPv6 host address about which to set permit conditions. The *source-ipv6-address*argument must be in the form documented in RFC 2373 where the address is specified in hexadecimal using 16-bit values between colons.

Verifying NAT-PT Configuration and Operation

SUMMARY STEPS

1. **clear ipv6 nat translation ***
2. **enable**
3. **debug ipv6 nat** [**detailed**| **port**]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**clear ipv6 nat translation *** **Example:** `Router> clear ipv6 nat translation *`	(Optional) Clears dynamic NAT-PT translations from the dynamic translation state table. • Use the ***** keyword to clear all dynamic NAT-PT translations. **Note** Static translation configuration is not affected by this command.

	Command or Action	Purpose
Step 2	**enable** **Example:** `Router> enable`	Enables higher privilege levels, such as privileged EXEC mode. • Enter your password if prompted.
Step 3	**debug ipv6 nat** [**detailed**| **port**] **Example:** `Router# debug ipv6 nat detail`	Displays debugging messages for NAT-PT translation events.

• Examples, page 468

Examples

Sample Output from the show ipv6 nat translations Command

In the following example, output information about active NAT-PT translations is displayed using the **show ipv6 nat translations**command:

```
Router# show ipv6 nat translations
Prot  IPv4 source              IPv6 source
      IPv4 destination         IPv6 destination
---   ---                      ---
      192.168.123.2            2001:DB8::2
---   ---                      ---
      192.168.122.10           2001:DB8::10
tcp   192.168.124.8,11047      2001:DB8:3::8,11047
      192.168.123.2,23         2001:DB8::2,23
udp   192.168.124.8,52922      2001:DB8:3::8,52922
      192.168.123.2,69         2001::2,69
udp   192.168.124.8,52922      2001:DB8:3::8,52922
      192.168.123.2,52922      2001:DB8::2,52922
---   192.168.124.8            2001:DB8:3::8
      192.168.123.2            2001:DB8::2
---   192.168.124.8            2001:DB8:3::8
      ---                      ---
---   192.168.121.4            2001:DB8:5::4
      ---                      ---
```

In the following example, detailed output information about active NAT-PT translations is displayed using the **show ipv6 nat translations**command with the **verbose** keyword:

```
Router# show ipv6 nat translations verbose
Prot  IPv4 source              IPv6 source
      IPv4 destination         IPv6 destination
---   ---                      ---
      192.168.123.2            2001:DB8::2
      create 00:04:24, use 00:03:24,
---   ---                      ---
      192.168.122.10           2001:DB8::10
      create 00:04:24, use 00:04:24,
tcp   192.168.124.8,11047      2001:DB8:3::8,11047
      192.168.123.2,23         2001:DB8::2,23
      create 00:03:24, use 00:03:20, left 00:16:39,
```

```
udp   192.168.124.8,52922      2001:DB8:3::8,52922
      192.168.123.2,69         2001:DB8::2,69
      create 00:02:51, use 00:02:37, left 00:17:22,
udp   192.168.124.8,52922      2001:DB8:3::8,52922
      192.168.123.2,52922      2001:DB8::2,52922
      create 00:02:48, use 00:02:30, left 00:17:29,
---   192.168.124.8            2001:DB8:3::8
      192.168.123.2            2001:DB8::2
      create 00:03:24, use 00:02:34, left 00:17:25,
---   192.168.124.8            2001:DB8:3::8
      ---                      ---
      create 00:04:24, use 00:03:24,
---   192.168.121.4            2001:DB8:5::4
      ---                      ---
      create 00:04:25, use 00:04:25,
```

Sample Output from the show ipv6 nat statistics Command

In the following example, output information about NAT-PT statistics is displayed using the **show ipv6 nat statistics**command:

```
Router# show ipv6 nat statistics
Total active translations: 4 (4 static, 0 dynamic; 0 extended)
NAT-PT interfaces:
  Ethernet3/1, Ethernet3/3
Hits: 0  Misses: 0
Expired translations: 0
```

Sample Output from the clear ipv6 nat translation Command

In the following example, all dynamic NAT-PT translations are cleared from the dynamic translation state table using the **clear ipv6 nat translation**command with the ***** keyword. When the output information about active NAT-PT translations is then displayed using the **show ipv6 nat translations**command, only the static translation configurations remain. Compare this **show** command output with the output from the **show ipv6 nat translations**command in Step 1.

```
Router# clear ipv6 nat translation *
Router# show ipv6 nat translations
Prot  IPv4 source              IPv6 source
      IPv4 destination         IPv6 destination
---   ---                      ---
      192.168.123.2            2001:DB8::2
---   ---                      ---
      192.168.122.10           2001:DB8::10
---   192.168.124.8            2001:DB8:3::8
      ---                      ---
---   192.168.121.4            2001:DB8:5::4
      ---                      ---
```

Sample Output from the debug ipv6 nat Command

In the following example, debugging messages for NAT-PT translation events are displayed using the **debug ipv6 nat**command:

```
Router# debug ipv6 nat
00:06:06: IPv6 NAT: icmp src (2001:DB8:3002::8) -> (192.168.124.8), dst
(2001:DB8:2001::2) -> (192.168.123.2)
00:06:06: IPv6 NAT: icmp src (192.168.123.2) -> (2001:DB8:2001::2), dst (192.168.124.8) -
> (2001:DB8:3002::8)
00:06:06: IPv6 NAT: icmp src (2001:DB8:3002::8) -> (192.168.124.8), dst
(2001:DB8:2001::2) -> (192.168.123.2)
00:06:06: IPv6 NAT: icmp src (192.168.123.2) -> (2001:DB8:2001::2), dst (192.168.124.8) -
> (2001:DB8:3002::8)
00:06:06: IPv6 NAT: tcp src (2001:DB8:3002::8) -> (192.168.124.8), dst (2001:DB8:2001::2)
-> (192.168.123.2)
00:06:06: IPv6 NAT: tcp src (192.168.123.2) -> (2001:DB8:2001::2), dst (192.168.124.8) ->
```

```
(2001:DB8:3002::8)
00:06:06: IPv6 NAT: tcp src (2001:DB8:3002::8) -> (192.168.124.8), dst (2001:DB8:2001::2)
-> (192.168.123.2)
00:06:06: IPv6 NAT: tcp src (2001:DB8:3002::8) -> (192.168.124.8), dst (2001:DB8:2001::2)
-> (192.168.123.2)
00:06:06: IPv6 NAT: tcp src (2001:DB8:3002::8) -> (192.168.124.8), dst (2001:DB8:2001::2)
-> (192.168.123.2)
00:06:06: IPv6 NAT: tcp src (192.168.123.2) -> (2001:DB8:2001::2), dst (192.168.124.8) ->
(2001:DB8:3002::8)
```

Configuration Examples for NAT-PT for IPv6

Example: Static NAT-PT Configuration

The following example configures the NAT-PT prefix globally, enables NAT-PT on two interfaces, and configures two static NAT-PT mappings. Ethernet interface 3/1 is configured as IPv6 only, and Ethernet interface 3/3 is configured as IPv4 only.

```
interface Ethernet3/1
 ipv6 address 2001:DB8:3002::9/64
 ipv6 enable
 ipv6 nat
!
interface Ethernet3/3
 ip address 192.168.30.9 255.255.255.0
 ipv6 nat
!
ipv6 nat v4v6 source 192.168.30.1 2001:DB8:0::2
ipv6 nat v6v4 source 2001:DB8:bbbb:1::1 10.21.8.10
ipv6 nat prefix 2001:DB8:0::/96
```

Example: Enabling Traffic to be Sent from an IPv6 Network to an IPv4 Network

In the following example, the access list permits any IPv6 source address with the prefix 2001::/96 to go to the destination with a 2000::/96 prefix. The destination is then translated to the last 32 bit of its IPv6 address; for example: source address = 2001::1, destination address = 2000::192.168.1.1. The destination then becomes 192.168.1.1 in the IPv4 network:

```
ipv6 nat prefix 2000::/96 v4-mapped v4map_acl
ipv6 access-list v4map_acl
 permit ipv6 2001::/96 2000::/96
```

Example: Dynamic NAT-PT Configuration for IPv6 Hosts Accessing IPv4 Hosts

The following example configures the NAT-PT prefix globally, enables NAT-PT on two interfaces, and configures one static NAT-PT mapping (used, for example, to access a DNS server). A dynamic NAT-PT mapping is also configured to map IPv6 addresses to IPv4 addresses using a pool of IPv4 addresses named

v4pool. The packets to be translated by NAT-PT are filtered using an IPv6 access list named pt-list1. The User Datagram Protocol (UDP) translation entries are configured to time out after 10 minutes. Ethernet interface 3/1 is configured as IPv6 only, and Ethernet interface 3/3 is configured as IPv4 only.

```
interface Ethernet3/1
 ipv6 address 2001:DB8:bbbb:1::9/64
 ipv6 enable
 ipv6 nat
!
interface Ethernet3/3
 ip address 192.168.30.9 255.255.255.0
 ipv6 nat
!
ipv6 nat v4v6 source 192.168.30.1 2001:DB8:0::2
ipv6 nat v6v4 source list pt-list1 pool v4pool
ipv6 nat v6v4 pool v4pool 10.21.8.1 10.21.8.10 prefix-length 24
ipv6 nat translation udp-timeout 600
ipv6 nat prefix 2001:DB8:1::/96
!
ipv6 access-list pt-list1
 permit ipv6 2001:DB8:bbbb:1::/64 any
```

Example Dynamic NAT-PT Configuration for IPv4 Hosts Accessing IPv6 Hosts

The following example configures the NAT-PT prefix globally, enables NAT-PT on two interfaces, and configures one static NAT-PT mapping (used, for example, to access a DNS server). A dynamic NAT-PT mapping is also configured to map IPv4 addresses to IPv6 addresses using a pool of IPv6 addresses named v6pool. The packets to be translated by NAT-PT are filtered using an access list named pt-list2. GigabitEthernet interface 3/1/1 is configured as IPv6 only, and GigabitEthernet interface 3/3/3 is configured as IPv4 only.

```
interface GigabitEthernet3/1/1
 ipv6 address 2001:DB8:bbbb:1::9/64
 ipv6 enable
 ipv6 nat
!
interface GigabitEthernet3/3/3
 ip address 192.168.30.9 255.255.255.0
 ipv6 nat
!
ipv6 nat v4v6 source list 72 pool v6pool
ipv6 nat v4v6 pool v6pool 2001:DB8:0::1 2001:DB8:0::2 prefix-length 128
ipv6 nat v6v4 source 2001:DB8:bbbb:1::1 10.21.8.0
ipv6 nat prefix 2001:DB8:0::/96
!
access-list 72 permit 192.168.30.0 0.0.0.255
```

Additional References

Related Documents

Related Topic	Document Title
IPv6 supported features	"Start Here: Cisco IOS XE Software Release Specifics for IPv6 Features ," *Cisco IOS XE IPv6 Configuration Guide*

Related Topic	Document Title
Basic IPv6 configuration tasks	" Implementing IPv6 Addressing and Basic Connectivity ," *Cisco IOS XE IPv6 Configuration Guide*
IPv6 commands: complete command syntax, command mode, defaults, usage guidelines, and examples	*Cisco IOS IPv6 Command Reference*

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
None	To locate and download MIBs for selected platforms, Cisco software releases, and feature sets, use Cisco MIB Locator found at the following URL: http://www.cisco.com/go/mibs

RFCs

RFCs	Title
RFC 2765	*Stateless IP/ICMP Translation Algorithm (SIIT)*
RFC 2766	*Network Address Translation - Protocol Translation (NAT-PT)*

Technical Assistance

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

Feature Information for Implementing NAT-PT for IPv6

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Table 20 Feature Information for Implementing NAT-PT for IPv6

Feature Name	Releases	Feature Information
NAT Protocol Translation	Cisco IOS XE Release 3.4S	NAT-PT is an IPv6-IPv4 translation mechanism that allows IPv6-only devices to communicate with IPv4-only devices and vice versa. NAT-PT is not supported in Cisco Express Forwarding.
NAT-PT--Support for DNS ALG	Cisco IOS XE Release 3.4S	IPv6 provides DNS ALG support.
NAT-PT--Support for FTP ALG	Cisco IOS XE Release 3.4S	IPv6 provides FTP ALG support.
NAT-PT--Support for Fragmentation	Cisco IOS XE Release 3.4S	Packet fragmentation is enabled by default when IPv6 is configured, allowing IPv6 and IPv4 networks to resolve fragmentation problems between the networks.
IPv6 Virtual Fragmentation Reassembly	Cisco IOS XE Release 3.4S	The IPv6 VFR feature provides the ability to collect the fragments and provide L4 info for all fragments for IPsec and NAT64 features.

Implementing OSPFv3

The *Implementing OSPF for IPv6* module expands on Open Shortest Path First version 3 (OSPFv3), or OSPF for IPv6, to provide support for IPv6 routing prefixes.

Finding Feature Information

Your software release may not support all the features documented in this module. For the latest caveats and feature information, see Bug Search Tool and the release notes for your platform and software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the feature information table at the end of this module.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Prerequisites for Implementing OSPFv3

- Complete the OSPFv3 network strategy and planning for your IPv6 network. For example, you must decide whether multiple areas are required.
- Enable IPv6 unicast routing.
- Enable IPv6 on the interface.
- Configure the IP security (IPsec) secure socket application program interface (API) on OSPFv3 in order to enable authentication and encryption.
- Before you can use the IPv4 unicast address families (AFs) in OSPFv3, you must enable IPv6 on a link, although the link may not be participating in IPv6 unicast AF.
- With the OSPFv3 Address Families feature, you may have two device processes per interface, but only one process per AF. If the AF is IPv4, you must first configure an IPv4 address on the interface, but IPv6 must be enabled on the interface.

Restrictions for Implementing OSPFv3

- When running a dual-stack IP network with OSPF version 2 for IPv4 and OSPFv3, be careful when changing the defaults for commands used to enable OSPFv3. Changing these defaults may affect your OSPFv3 network, possibly adversely.
- A packet will be rejected on a router if the packet is coming from an IPv6 address that is found on any interface on the same router.

Information About Implementing OSPFv3

How OSPFv3 Works

OSPFv3 is a routing protocol for IPv4 and IPv6. It is a link-state protocol, as opposed to a distance-vector protocol. Think of a link as being an interface on a networking device. A link-state protocol makes its routing decisions based on the states of the links that connect source and destination machines. The state of a link is a description of that interface and its relationship to its neighboring networking devices. The interface information includes the IPv6 prefix of the interface, the network mask, the type of network it is connected to, the devices connected to that network, and so on. This information is propagated in various type of link-state advertisements (LSAs).

A device's collection of LSA data is stored in a link-state database. The contents of the database, when subjected to the Dijkstra algorithm, result in the creation of the OSPF routing table. The difference between the database and the routing table is that the database contains a complete collection of raw data; the routing table contains a list of shortest paths to known destinations via specific device interface ports.

OSPFv3, which is described in RFC 5340, supports IPv6 and IPv4 unicast AFs.

Comparison of OSPFv3 and OSPF Version 2

Much of the OSPFv3 feature is the same as in OSPF version 2. OSPFv3, which is described in RFC 5340, expands on OSPF version 2 to provide support for IPv6 routing prefixes and the larger size of IPv6 addresses.

In OSPFv3, a routing process does not need to be explicitly created. Enabling OSPFv3 on an interface will cause a routing process, and its associated configuration, to be created.

In OSPFv3, each interface must be enabled using commands in interface configuration mode. This feature is different from OSPF version 2, in which interfaces are indirectly enabled using the router configuration mode.

In IPv6, users can configure many address prefixes on an interface. In OSPFv3, all address prefixes on an interface are included by default. Users cannot select some address prefixes to be imported into OSPFv3; either all address prefixes on an interface are imported, or no address prefixes on an interface are imported.

Unlike OSPF version 2, multiple instances of OSPFv3 can be run on a link.

In OSPFv3, it is possible that no IPv4 addresses will be configured on any interface. In this case, the user must use the router-id command to configure a router ID before the OSPF process will be started. A router ID is a 32-bit opaque number. OSPF version 2 takes advantage of the 32-bit IPv4 address to pick an IPv4 address as the router ID. If an IPv4 address does exist when OSPFv3 is enabled on an interface, then that IPv4 address is used for the router ID. If more than one IPv4 address is available, a router ID is chosen using the same rules as for OSPF version 2.

OSPF automatically prefers a loopback interface over any other kind, and it chooses the highest IP address among all loopback interfaces. If no loopback interfaces are present, the highest IP address in the router is chosen. You cannot tell OSPF to use any particular interface.

OSPFv3 Address Families

The OSPFv3 address families feature enables both IPv4 and IPv6 unicast traffic to be supported. With this feature, users may have two router processes per interface, but only one process per AF. If the IPv4 AF is used, an IPv4 address must first be configured on the interface, but IPv6 must be enabled on the interface. A single IPv4 or IPv6 OSPFv3 process running multiple instances on the same interface is not supported.

Users with an IPv6 network that uses OSPFv3 as its IGP may want to use the same IGP to help carry and install IPv4 routes. All routers on this network have an IPv6 forwarding stack. Some (or all) of the links on this network may be allowed to do IPv4 forwarding and be configured with IPv4 addresses. Pockets of IPv4-only routers exist around the edges running an IPv4 static or dynamic routing protocol. In this scenario, users need the ability to forward IPv4 traffic between these pockets without tunneling overhead, which means that any IPv4 transit router has both IPv4 and IPv6 forwarding stacks (e.g., is dual stack).

This feature allows a separate (possibly incongruent) topology to be constructed for the IPv4 AF. It installs IPv4 routes in IPv4 RIB, and then the forwarding occurs natively. The OSPFv3 process fully supports an IPv4 AF topology and can redistribute routes from and into any other IPv4 routing protocol.

An OSPFv3 process can be configured to be either IPv4 or IPv6. The **address-family** command is used to determine which AF will run in the OSPFv3 process, and only one address family can be configured per instance. Once the AF is selected, users can enable multiple instances on a link and enable address-family-specific commands.

Different instance ID ranges are used for each AF. Each AF establishes different adjacencies, has a different link state database, and computes a different shortest path tree. The AF then installs the routes in AF-specific RIB. LSAs that carry IPv6 unicast prefixes are used without any modification in different instances to carry each AFs' prefixes.

The IPv4 subnets configured on OSPFv3-enabled interfaces are advertised through intra-area prefix LSAs, just as any IPv6 prefixes. External LSAs are used to advertise IPv4 routes redistributed from any IPv4 routing protocol, including connected and static. The IPv4 OSPFv3 process runs the SPF calculations and finds the shortest path to those IPv4 destinations. These computed routes are then inserted in the IPv4 RIB (computed routes are inserted into an IPv6 RIB for an IPv6 AF).

Because the IPv4 OSPFv3 process allocates a unique pdbindex in the IPv4 RIB, all other IPv4 routing protocols can redistribute routes from it. The parse chain for all protocols is same, so the **ospfv3** keyword added to the list of IPv4 routing protocols causes OSPFv3 to appear in the **redistribute** command from any

IPv4 routing protocol. With the **ospfv3** keyword, IPv4 OSPFv3 routes can be redistributed into any other IPv4 routing protocol as defined in the **redistribute ospfv3** command.

The OSPFv3 address families feature is supported as of Cisco IOS XE Release 3.4S. Cisco routers that run software older than this release and third-party routers will not neighbor with routers running the AF feature for the IPv4 AF because they do not set the AF bit. Therefore, those routers will not participate in the IPv4 AF SPF calculations and will not install the IPv4 OSPFv3 routes in the IPv6 RIB.

LSA Types for OSPFv3

The following list describes LSA types, each of which has a different purpose:

- Router LSAs (Type 1)—Describes the link state and costs of a router's links to the area. These LSAs are flooded within an area only. The LSA indicates if the router is an Area Border Router (ABR) or Autonomous System Boundary Router (ASBR), and if it is one end of a virtual link. Type 1 LSAs are also used to advertise stub networks. In OSPFv3, these LSAs have no address information and are network-protocol-independent. In OSPFv3, router interface information may be spread across multiple router LSAs. Receivers must concatenate all router LSAs originated by a given router when running the SPF calculation.
- Network LSAs (Type 2)—Describes the link-state and cost information for all routers attached to the network. This LSA is an aggregation of all the link-state and cost information in the network. Only a designated router tracks this information and can generate a network LSA. In OSPFv3, network LSAs have no address information and are network-protocol-independent.
- Interarea-prefix LSAs for ABRs (Type 3)—Advertises internal networks to routers in other areas (interarea routes). Type 3 LSAs may represent a single network or a set of networks summarized into one advertisement. Only ABRs generate summary LSAs. In OSPFv3, addresses for these LSAs are expressed as *prefix, prefix length* instead of *address, mask*. The default route is expressed as a prefix with length 0.
- Interarea-router LSAs for ASBRs (Type 4)—Advertises the location of an ASBR. Routers that are trying to reach an external network use these advertisements to determine the best path to the next hop. Type 4 LSAs are generated by ABRs on behalf of ASBRs.
- Autonomous system external LSAs (Type 5)—Redistributes routes from another autonomous system, usually from a different routing protocol into OSPFv3. In OSPFv3, addresses for these LSAs are expressed as *prefix, prefix length* instead of *address, mask*. The default route is expressed as a prefix with length 0.
- Link LSAs (Type 8)—Have local-link flooding scope and are never flooded beyond the link with which they are associated. Link LSAs provide the link-local address of the router to all other routers attached to the link, inform other routers attached to the link of a list of prefixes to associate with the link, and allow the router to assert a collection of Options bits to associate with the network LSA that will be originated for the link.
- Intra-Area-Prefix LSAs (Type 9)—A router can originate multiple intra-area-prefix LSAs for each router or transit network, each with a unique link-state ID. The link-state ID for each intra-area-prefix LSA describes its association to either the router LSA or the network LSA and contains prefixes for stub and transit networks.

An address prefix occurs in almost all newly defined LSAs. The prefix is represented by three fields: PrefixLength, PrefixOptions, and Address Prefix. In OSPFv3, addresses for these LSAs are expressed as *prefix, prefix length* instead of *address, mask.* The default route is expressed as a prefix with length 0. Type 3 and Type 9 LSAs carry all prefix (subnet) information that, in OSPFv2, is included in router LSAs and network LSAs. The Options field in certain LSAs (router LSAs, network LSAs, interarea-router LSAs, and link LSAs) has been expanded to 24 bits to provide support for OSPFv3.

In OSPFv3, the sole function of the link-state ID in interarea-prefix LSAs, interarea-router LSAs, and autonomous-system external LSAs is to identify individual pieces of the link-state database. All addresses

or router IDs that are expressed by the link-state ID in OSPF version 2 are carried in the body of the LSA in OSPFv3.

The link-state ID in network LSAs and link LSAs is always the interface ID of the originating router on the link being described. For this reason, network LSAs and link LSAs are now the only LSAs whose size cannot be limited. A network LSA must list all routers connected to the link, and a link LSA must list all of the address prefixes of a router on the link.

- OSPFv3 Max-Metric Router LSA, page 479

OSPFv3 Max-Metric Router LSA

The OSPFv3 max-metric router LSA feature enables OSPFv3 to advertise its locally generated router LSAs with a maximum metric. The feature allows OSPFv3 processes to converge but not attract transit traffic through the device if there are better alternate paths. After a specified timeout or a notification from Border Gateway Protocol (BGP), OSPFv3 advertises the LSAs with normal metrics.

The max-metric LSA control places the OSPFv3 router into the stub router role using its LSA advertisement. A stub router only forwards packets destined to go to its directly connected links. In OSPFv3 networks, a device could become a stub router by advertising large metrics for its connected links, so that the cost of a path through this device becomes larger than that of an alternative path. OSPFv3 stub router advertisement allows a device to advertise the infinity metric (0xFFFF) for its connected links in router LSAs and advertise the normal interface cost if the link is a stub network.

Fast Convergence: LSA and SPF Throttling

The OSPFv3 LSA and SPF throttling feature provides a dynamic mechanism to slow down link-state advertisement updates in OSPFv3 during times of network instability. It also allows faster OSPFv3 convergence by providing LSA rate limiting in milliseconds.

OSPFv3 can use static timers for rate-limiting SPF calculation and LSA generation. Although these timers are configurable, the values used are specified in seconds, which poses a limitation on OSPFv3 convergence. LSA and SPF throttling achieves subsecond convergence by providing a more sophisticated SPF and LSA rate-limiting mechanism that is able to react quickly to changes and also provide stability and protection during prolonged periods of instability.

Addresses Imported into OSPFv3

When importing the set of addresses specified on an interface on which OSPFv3 is running into OSPFv3, you cannot select specific addresses to be imported. Either all addresses are imported, or no addresses are imported.

OSPFv3 Authentication Support with IPsec

In order to ensure that OSPFv3 packets are not altered and re-sent to the router, causing the router to behave in a way not desired by its managers, OSPFv3 packets must be authenticated. OSPFv3 uses the IP Security (IPsec) secure socket application program interface (API) to add authentication to OSPFv3 packets. This API has been extended to provide support for IPv6.

OSPFv3 requires the use of IPsec to enable authentication. Crypto images are required to use authentication, because only crypto images include the IPsec API needed for use with OSPFv3.

In OSPFv3, authentication fields have been removed from OSPFv3 packet headers. When OSPFv3 runs on IPv6, OSPFv3 requires the IPv6 authentication header (AH) or IPv6 ESP header to ensure integrity,

authentication, and confidentiality of routing exchanges. IPv6 AH and ESP extension headers can be used to provide authentication and confidentiality to OSPFv3.

To use the IPsec AH, you must enable the **ipv6 ospf authentication** command. To use the IPsec ESP, you must enable the **ipv6 ospf encryption** command. The ESP header may be applied alone or in combination with the AH, and when ESP is used, both encryption and authentication are provided. Security services can be provided between a pair of communicating hosts, between a pair of communicating security gateways, or between a security gateway and a host.

To configure IPsec, users configure a security policy, which is a combination of the security policy index (SPI) and the key (the key is used to create and validate the hash value). IPsec for OSPFv3 can be configured on an interface or on an OSPFv3 area. For higher security, users should configure a different policy on each interface configured with IPsec. If a user configures IPsec for an OSPFv3 area, the policy is applied to all of the interfaces in that area, except for the interfaces that have IPsec configured directly. Once IPsec is configured for OSPFv3, IPsec is invisible to the user.

The secure socket API is used by applications to secure traffic. The API needs to allow the application to open, listen, and close secure sockets. The binding between the application and the secure socket layer also allows the secure socket layer to inform the application of changes to the socket, such as connection open and close events. The secure socket API is able to identify the socket; that is, it can identify the local and remote addresses, masks, ports, and protocol that carry the traffic requiring security.

Each interface has a secure socket state, which can be one of the following:

- NULL: Do not create a secure socket for the interface if authentication is configured for the area.
- DOWN: IPsec has been configured for the interface (or the area that contains the interface), but OSPFv3 either has not requested IPsec to create a secure socket for this interface, or there is an error condition.
- GOING UP: OSPFv3 has requested a secure socket from IPsec and is waiting for a CRYPTO_SS_SOCKET_UP message from IPsec.
- UP: OSPFv3 has received a CRYPTO_SS_SOCKET_UP message from IPsec.
- CLOSING: The secure socket for the interface has been closed. A new socket may be opened for the interface, in which case the current secure socket makes the transition to the DOWN state. Otherwise, the interface will become UNCONFIGURED.
- UNCONFIGURED: Authentication is not configured on the interface.

OSPFv3 will not send or accept packets while in the DOWN state.

- OSPFv3 Virtual Links, page 480
- OSPFv3 Cost Calculation, page 480

OSPFv3 Virtual Links

For each virtual link, a master security information datablock is created for the virtual link. Because a secure socket must be opened on each interface, there will be a corresponding security information datablock for each interface in the transit area. The secure socket state is kept in the interface's security information datablock. The state field in the master security information datablock reflects the status of all of the secure sockets opened for the virtual link. If all of the secure sockets are UP, then the security state for the virtual link will be set to UP.

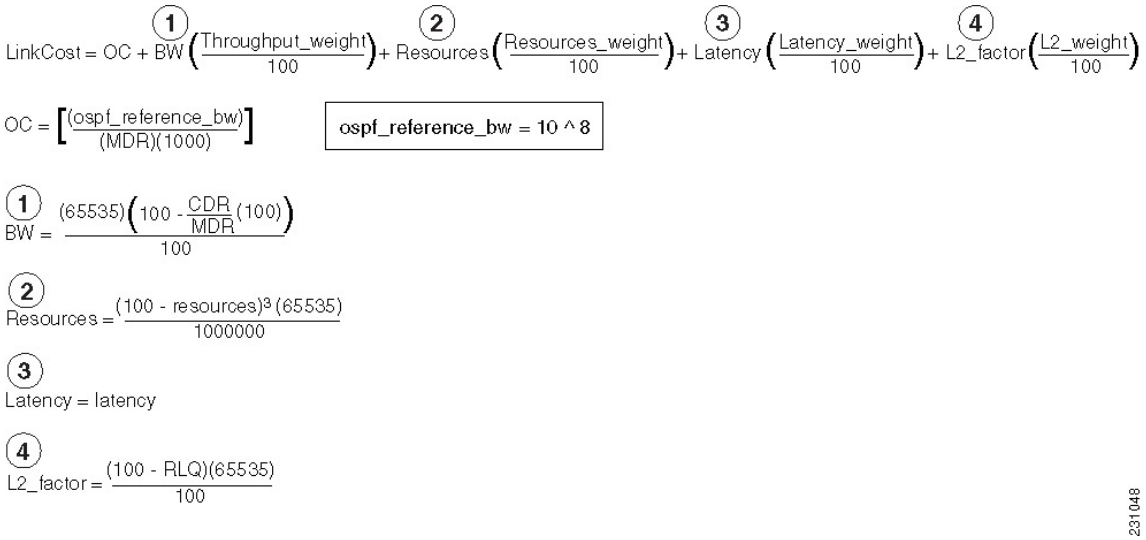
OSPFv3 Cost Calculation

Because cost components can change rapidly, it might be necessary to reduce the volume of changes to reduce network-wide churn. The recommended values for S2, S3, and S4 in the second table below are

based on network simulations that may reduce the rate of network changes. The recommended value for S1 is 0 to eliminate this variable from the route cost calculation.

The overall link cost is computed using the formula shown in the figure below.

Figure 43 Overall Link Cost Formula



The table below defines the symbols used in the OSPFv3 cost calculation.

Table 21 OSPFv3 Cost Calculation Definitions

Cost Component	Component Definition
OC	The default OSPFv3 cost. Calculated from reference bandwidth using reference_bw / (MDR*1000), where reference_bw=10^8.
A through D	Various radio-specific data-based formulas that produce results in the 0 through 64,000 range.
A	CDR- and MDR-related formula: (2^16 * (100 – (CDR * 100 / MDR)))/100
B	Resources related formula: ((100 – RESOURCES)^3 * 2^16 / 10^6)
C	Latency as reported by the radio, already in the 0 through 64,000 range when reported (LATENCY).
D	RLF-related formula: ((100 – RLF) * 2^16)/100

Cost Component	Component Definition
S1 through S4	Scalar weighting factors input from the CLI. These scalars scale down the values as computed by A through D. The value of 0 disables and the value of 100 enables full 0 through 64,000 range for one component.

Because each network might have unique characteristics that require different settings to optimize actual network performance, these are recommended values intended as a starting point for optimizing an OSPFv3 network. The table below lists the recommended value settings for OSPFv3 cost metrics.

Table 22 Recommended Value Settings for OSPFv3 Cost Metrics

Setting	Metric Description	Default Value	Recommended Value
S1	ipv6 ospf dynamic weight throughout	100	0
S2	ipv6 ospf dynamic weight resources	100	29
S3	ipv6 ospf dynamic weight latency	100	29
S4	ipv6 ospf dynamic weight L2 factor	100	29

The default path costs were calculated using this formula, as noted in the following list. If these values do not suit your network, you can use your own method of calculating path costs.

- 56-kbps serial link—Default cost is 1785.
- 64-kbps serial link—Default cost is 1562.
- T1 (1.544-Mbps serial link)—Default cost is 64.
- E1 (2.048-Mbps serial link)—Default cost is 48.
- 4-Mbps Token Ring—Default cost is 25.
- Ethernet—Default cost is 10.
- 16-Mbps Token Ring—Default cost is 6.
- FDDI—Default cost is 1.
- X25—Default cost is 5208.
- Asynchronous—Default cost is 10,000.
- ATM—Default cost is 1.

To illustrate these settings, the following example shows how OSPFv3 cost metrics might be defined for a Virtual Multipoint Interface (VMI) interface:

```
interface vmi1
 ipv6 ospf cost dynamic weight throughput 0
 ipv6 ospf cost dynamic weight resources 29
 ipv6 ospf cost dynamic weight latency 29
 ipv6 ospf cost dynamic weight L2-factor 29
```

OSPFv3 Customization

You can customize OSPFv3 for your network, but you likely will not need to do so. The defaults for OSPFv3 are set to meet the requirements of most customers and features. If you must change the defaults, refer to the IPv6 command reference to find the appropriate syntax.

Caution Be careful when changing the defaults. Changing defaults will affect your OSPFv3 network, possibly adversely.

- OSPFv3 Virtual Links, page 483

OSPFv3 Virtual Links

For each virtual link, a master security information datablock is created for the virtual link. Because a secure socket must be opened on each interface, there will be a corresponding security information datablock for each interface in the transit area. The secure socket state is kept in the interface's security information datablock. The state field in the master security information datablock reflects the status of all of the secure sockets opened for the virtual link. If all of the secure sockets are UP, then the security state for the virtual link will be set to UP.

Link Quality Metrics Reporting for OSPFv3 with VMI Interfaces

OSPFv3 is one of the routing protocols that can be used with Virtual Multipoint Interfaces (VMIs) in router-to-radio networks. The quality of a radio link has a direct impact on the throughput that can be achieved by router-router traffic. The PPPoE protocol has been extended to provide a process by which a router can request, or a radio can report, link quality metric information. Cisco's OSFPv3 implementation has been enhanced so that the route cost to a neighbor is dynamically updated based on metrics reported by the radio, thus allowing the best route to be chosen within a given set of radio links.

The routing protocols receive raw radio link data, and compute a composite quality metric for each link. In computing these metrics, the following factors may be considered:

- Maximum Data Rate--the theoretical maximum data rate of the radio link, in bytes per second
- Current Data Rate--the current data rate achieved on the link, in bytes per second
- Latency--the transmission delay packets encounter, in milliseconds
- Resources--a percentage (0 to 100) that can represent the remaining amount of a resource (such as battery power)
- Relative Link Quality--a numeric value (0-100) representing relative quality, with 100 being the highest quality

Metrics can be weighted during the configuration process to emphasize or de-emphasize particular characteristics. For example, if throughput is a particular concern, the current data rate metric could be weighted so that it is factored more heavily into the composite metric. Similarly, a metric that is of no concern can be omitted from the composite calculation.

Link metrics can change rapidly, often by very small degrees, which could result in a flood of meaningless routing updates. In a worst case scenario, the network would be churning almost continuously as it struggled to react to minor variations in link quality. To alleviate this concern, Cisco provides a tunable dampening mechanism that allows the user to configure threshold values. Any metric change that falls below the threshold is ignored.The quality of a connection to a neighbor varies, based on various

characteristics of the interface when OSPF is used as the routing protocol. The routing protocol receives dynamic raw radio link characteristics and computes a composite metric that is used to reduce the effect of frequent routing changes.

A tunable hysteresis mechanism allows users to adjust the threshold to the routing changes that occur when the router receives a signal that a new peer has been discovered, or that an existing peer is unreachable. The tunable metric is weighted and is adjusted dynamically to account for the following characteristics:

- Current and maximum bandwidth
- Latency
- Resources
- L2 factor

Individual weights can be deconfigured and all weights can be cleared so that the cost is set back to the default value for the interface type. Based on the routing changes that occur, cost can be determined by the application of these metrics.

OSPFv3 External Path Preference Option

Per RFC 5340, the following rules indicate which paths are preferred when multiple intra-AS paths are available to ASBRs or forwarding addresses:

- Intra-area paths using nonbackbone areas are always the most preferred.
- The other paths, intraarea backbone paths and interarea paths, are of equal preference.

These rules apply when the same ASBR is reachable through multiple areas, or when trying to decide which of several AS-external-LSAs should be preferred. In the former case the paths all terminate at the same ASBR, and in the latter the paths terminate at separate ASBRs or forwarding addresses. In either case, each path is represented by a separate routing table entry. This feature applies only when RFC 1583 compatibility is set to disabled using the **no compatibility rfc1583** command (RFC 5340 provides an update to RFC 1583).

Caution To minimize the chance of routing loops, set identical RFC compatibility for all OSPF routers in an OSPF routing domain.

OSPFv3 Graceful Restart

The graceful restart feature in OSPFv3 allows nonstop data forwarding along routes that are already known while the OSPFv3 routing protocol information is being restored. A device can participate in graceful restart either in restart mode (such as in a graceful-restart-capable router) or in helper mode (such as in a graceful-restart-aware router).

To perform the graceful restart function, a device must be in high availability (HA) stateful switchover (SSO) mode (that is, dual Route Processor (RP)). A device capable of graceful restart will perform the graceful restart function when the following failures occur:

- A RP failure that results in switchover to standby RP
- A planned RP switchover to standby RP

The graceful restart feature requires that neighboring devices be graceful-restart aware.

For further information about SSO and nonstop forwarding (NSF), see the Stateful Switchover and Cisco Nonstop Forwarding documents.

How to Implement OSPFv3

Configuring the OSPFv3 Router Process

Once you have completed step 3 and entered OSPFv3 router configuration mode, you can perform any of the subsequent steps in this task as needed to configure OSPFv3 router configuration.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **router ospfv3** [*process-id*]
4. **area** *area-ID* [**default-cost** | **nssa** | **stub**]
5. **auto-cost reference-bandwidth** *Mbps*
6. **bfd all-interfaces**
7. **default** {**area** *area-ID*[**range** *ipv6-prefix* | **virtual-link** *router-id*]} [**default-information originate** [**always** | **metric** | **metric-type** | **route-map**] | **distance** | **distribute-list** *prefix-list prefix-list-name* {**in** | **out**} [*interface*] | **maximum-paths** *paths* | **redistribute** *protocol* | **summary-prefix** *ipv6-prefix*]
8. **ignore lsa mospf**
9. **interface-id snmp-if-index**
10. **log-adjacency-changes** [**detail**]
11. passive-interface [**default** | *interface-type interface-number*]
12. **queue-depth** {**hello** | **update**} {*queue-size* | **unlimited**}
13. router-id {*router-id*}

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**router ospfv3** [*process-id*] **Example:** `Router(config)# router ospfv3 1`	Enables OSPFv3 router configuration mode for the IPv4 or IPv6 address family.
Step 4	**area** *area-ID* [**default-cost** | **nssa** | **stub**] **Example:** `Router(config-router)# area 1`	Configures the OSPFv3 area.
Step 5	**auto-cost reference-bandwidth** *Mbps* **Example:** `Router(config-router)# auto-cost reference-bandwidth 1000`	Controls the reference value OSPFv3 uses when calculating metrics for interfaces in an IPv4 OSPFv3 process.
Step 6	**bfd all-interfaces** **Example:** `Router(config-router)# bfd all-interfaces`	Enables BFD for an OSPFv3 routing process

	Command or Action	Purpose
Step 7	**default** {**area** *area-ID*[**range** *ipv6-prefix* | **virtual-link** *router-id*]} [**default-information originate** [**always** | **metric** | **metric-type** | **route-map**] | **distance** | **distribute-list** *prefix-list prefix-list-name* {**in** | **out**} [*interface*] | **maximum-paths** *paths* | **redistribute** *protocol* | **summary-prefix** *ipv6-prefix*] **Example:** `Router(config-router)# default area 1`	Returns an OSPFv3 parameter to its default value.
Step 8	**ignore lsa mospf** **Example:** `Router(config-router)# ignore lsa mospf`	Suppresses the sending of syslog messages when the router receives LSA Type 6 multicast OSPFv3 packets, which are unsupported.
Step 9	**interface-id snmp-if-index** **Example:** `Router(config-router)# interface-id snmp-if-index`	Configures OSPFv3 interfaces with Simple Network Management Protocol (SNMP) MIB-II interface Index (ifIndex) identification numbers in IPv4 and IPv6.
Step 10	**log-adjacency-changes** [**detail**] **Example:** `Router(config-router)# log-adjacency-changes`	Configures the router to send a syslog message when an OSPFv3 neighbor goes up or down.
Step 11	passive-interface [**default** | *interface-type interface-number*] **Example:** `Router(config-router)# passive-interface default`	Suppresses sending routing updates on an interface when using an IPv4 OSPFv3 process.
Step 12	**queue-depth** {**hello** | **update**} {*queue-size* | **unlimited**} **Example:** `Router(config-router)# queue-depth update 1500`	Configures the number of incoming packets that the IPv4 OSPFv3 process can keep in its queue.
Step 13	router-id {*router-id*} **Example:** `Router(config-router)# router-id 10.1.1.1`	Use a fixed router ID.

Configuring the IPv6 Address Family in OSPFv3

Perform this task to configure the IPv6 address family in OSPFv3. Once you have completed step 4 and entered IPv6 address-family configuration mode, you can perform any of the subsequent steps in this task as needed to configure the IPv6 AF.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **router ospfv3** [*process-id*]
4. **address-family ipv6 unicast**
5. **area** *area-ID* **range** *ipv6-prefix* / *prefix-length*
6. **default** {**area** *area-ID*[**range** *ipv6-prefix* | **virtual-link** *router-id*]} [**default-information originate** [**always** | **metric** | **metric-type** | **route-map**] | **distance** | **distribute-list** *prefix-list prefix-list-name* {**in** | **out**} [*interface*] | **maximum-paths** *paths* | **redistribute** *protocol* | **summary-prefix** *ipv6-prefix*]
7. **default-information originate** [**always** | **metric** *metric-value* | **metric-type** *type-value*| **route-map** *map-name*]
8. default-metric *metric-value*
9. **distance** *distance*
10. **distribute-list prefix-list** *list-name* {**in**[*interface-type interface-number*] | **out** *routing-process* [*as-number*]}
11. **maximum-paths** *number-paths*
12. **summary-prefix** *prefix* [**not-advertise** | **tag** *tag-value*]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**router ospfv3** [*process-id*] **Example:** `Router(config)# router ospfv3 1`	Enables OSPFv3 router configuration mode for the IPv4 or IPv6 address family.

	Command or Action	Purpose

Step 4	**address-family ipv6 unicast**
	
	Example:
	
	Example:
	or
	
	Example:
	
	address-family ipv4
	unicast
	
	Example:
	`Router(config-router)# address-family ipv6 unicast`
	
	Example:
	
	Example:
	or
	
	Example:
	`Router(config-router)# address-family ipv4 unicast`
		Enters IPv6 address family configuration mode for OSPFv3.
		or
		Enters IPv4 address family configuration mode for OSPFv3.

Step 5	**area** *area-ID* **range** *ipv6-prefix* / *prefix-length* **Example:** `Router(config-router-af)# area 1 range 2001:DB8:0:0::0/128`	Configures OSPFv3 area parameters.

	Command or Action	Purpose
Step 6	**default** {**area** *area-ID*[**range** *ipv6-prefix* | **virtual-link** *router-id*]} [**default-information originate** [**always** | **metric** | **metric-type** | **route-map**] | **distance** | **distribute-list** *prefix-list prefix-list-name* {**in** | **out**} [*interface*] | **maximum-paths** *paths* | **redistribute** *protocol* | **summary-prefix** *ipv6-prefix*] **Example:** `Router(config-router-af)# default area 1`	Returns an OSPFv3 parameter to its default value.
Step 7	**default-information originate** [**always** | **metric** *metric-value* | **metric-type** *type-value*| **route-map** *map-name*] **Example:** `Router(config-router-af)# default-information originate always metric 100 metric-type 2`	Generates a default external route into an OSPFv3 for a routing domain.
Step 8	default-metric *metric-value* **Example:** `Router(config-router-af)# default-metric 10`	Sets default metric values for IPv4 and IPv6 routes redistributed into the OSPFv3 routing protocol.
Step 9	**distance** *distance* **Example:** `Router(config-router-af)# distance 200`	Configures an administrative distance for OSPFv3 routes inserted into the routing table.
Step 10	**distribute-list prefix-list** *list-name* {**in**[*interface-type interface-number*] | **out** *routing-process* [*as-number*]} **Example:** `Router(config-router-af)# distribute-list prefix-list PL1 in Ethernet0/0`	Applies a prefix list to OSPFv3 routing updates that are received or sent on an interface.
Step 11	**maximum-paths** *number-paths* **Example:** `Router(config-router-af)# maximum-paths 4`	Controls the maximum number of equal-cost routes that a process for OSPFv3 routing can support.

	Command or Action	Purpose
Step 12	**summary-prefix** *prefix* [**not-advertise** | **tag** *tag-value*] **Example:** `Router(config-router-af)# summary-prefix FEC0::/24`	Configures an IPv6 summary prefix in OSPFv3.

Configuring the IPv4 Address Family in OSPFv3

Perform this task to configure the IPv4 address family in OSPFv3. Once you have completed step 4 and entered IPv4 address-family configuration mode, you can perform any of the subsequent steps in this task as needed to configure the IPv4 AF.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **router ospfv3** [*process-id*]
4. **address-family ipv4 unicast**
5. **area** *area-id* **range** *ip-address ip-address-mask* [**advertise** | **not-advertise**] [**cost** *cost*]
6. **default** {**area** *area-ID*[**range** *ipv6-prefix* | **virtual-link** *router-id*]} [**default-information originate** [**always** | **metric** | **metric-type** | **route-map**] | **distance** | **distribute-list** *prefix-list prefix-list-name* {**in** | **out**} [*interface*] | **maximum-paths** *paths* | **redistribute** *protocol* | **summary-prefix** *ipv6-prefix*]
7. **default-information originate** [**always** | **metric** *metric-value* | **metric-type** *type-value*| **route-map** *map-name*]
8. default-metric *metric-value*
9. **distance** *distance*
10. **distribute-list prefix-list** *list-name* {**in**[*interface-type interface-number*] | **out** *routing-process* [*as-number*]}
11. **maximum-paths** *number-paths*
12. **summary-prefix** *prefix* [**not-advertise** | **tag** *tag-value*]

DETAILED STEPS

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

	Command or Action	Purpose
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**router ospfv3** [*process-id*] **Example:** `Router(config)# router ospfv3 1`	Enables OSPFv3 router configuration mode for the IPv4 or IPv6 address family.
Step 4	**address-family ipv4 unicast** **Example:** `Router(config-router)# address-family ipv4 unicast`	Enters IPv4 address family configuration mode for OSPFv3.
Step 5	**area** *area-id* **range** *ip-address ip-address-mask* [**advertise** | **not-advertise**] [**cost** *cost*] **Example:** `Router(config-router-af)# area 0 range 192.168.110.0 255.255.0.0`	Consolidates and summarizes routes at an area boundary.
Step 6	**default** {**area** *area-ID*[**range** *ipv6-prefix* | **virtual-link** *router-id*]} [**default-information originate** [**always** | **metric** | **metric-type** | **route-map**] | **distance** | **distribute-list** *prefix-list prefix-list-name* {**in** | **out**} [*interface*] | **maximum-paths** *paths* | **redistribute** *protocol* | **summary-prefix** *ipv6-prefix*] **Example:** `Router(config-router-af)# default area 1`	Returns an OSPFv3 parameter to its default value.
Step 7	**default-information originate** [**always** | **metric** *metric-value* | **metric-type** *type-value*| **route-map** *map-name*] **Example:** `Router(config-router-af)# default-information originate always metric 100 metric-type 2`	Generates a default external route into an OSPFv3 for a routing domain.

	Command or Action	Purpose
Step 8	default-metric *metric-value* **Example:** `Router(config-router-af)# default-metric 10`	Sets default metric values for IPv4 and IPv6 routes redistributed into the OSPFv3 routing protocol.
Step 9	**distance** *distance* **Example:** `Router(config-router-af)# distance 200`	Configures an administrative distance for OSPFv3 routes inserted into the routing table.
Step 10	**distribute-list prefix-list** *list-name* {**in**[*interface-type interface-number*] | **out** *routing-process* [*as-number*]} **Example:** `Router(config-router-af)# distribute-list prefix-list PL1 in Ethernet0/0`	Applies a prefix list to OSPFv3 routing updates that are received or sent on an interface.
Step 11	**maximum-paths** *number-paths* **Example:** `Router(config-router-af)# maximum-paths 4`	Controls the maximum number of equal-cost routes that a process for OSPFv3 routing can support.
Step 12	**summary-prefix** *prefix* [**not-advertise** | **tag** *tag-value*] **Example:** `Router(config-router-af)# summary-prefix FEC0::/24`	Configures an IPv6 summary prefix in OSPFv3.

Configuring Route Redistribution in OSPFv3

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **router ospfv3** [*process-id*]
4. **address-family ipv6 unicast**
5. **redistribute** source-*protocol* [*process-id*] [*options*]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**router ospfv3** [*process-id*] **Example:** `Router(config)# router ospfv3 1`	Enables OSPFv3 router configuration mode for the IPv4 or IPv6 address family.

	Command or Action	Purpose

Step 4	**address-family ipv6 unicast**
	
	Example:
	
	Example:
	or
	
	Example:
	address-family ipv4
	unicast
	
	Example:
	`Router(config-router)# address-family ipv6 unicast`
	
	Example:
	
	Example:
	or
	
	Example:
	`Router(config-router)# address-family ipv4 unicast`
		Enters IPv6 address family configuration mode for OSPFv3.
		or
		Enters IPv4 address family configuration mode for OSPFv3.

Step 5	**redistribute** source-*protocol* [*process-id*] [*options*] **Example:**	Redistributes IPv6 and IPv4 routes from one routing domain into another routing domain.

Enabling OSPFv3 on an Interface

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type number*
4. Do one of the following:
   - **ospfv3** *process-id* **area** *area-ID* {**ipv4** | **ipv6**} [**instance** *instance-id*]
   - **ipv6 ospf** *process-id* **area** *area-id* [**instance** *instance-id*]

DETAILED STEPS

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

Step 4	Do one of the following:
	• **ospfv3** *process-id* **area** *area-ID* {**ipv4** | **ipv6**} [**instance** *instance-id*]
	• **ipv6 ospf** *process-id* **area** *area-id* [**instance** *instance-id*]
	
	Example:
	`Device(config-if)# ospfv3 1 area 1 ipv4`
	
	Example:
	`Device(config-if)# ipv6 ospf 1 area 0`
		Enables OSPFv3 on an interface with the IPv4 or IPv6 AF.
		or
		Enables OSPFv3 on an interface.

Defining an OSPFv3 Area Range for the IPv6 or IPv4 Address Family

The cost of the summarized routes will be the highest cost of the routes being summarized. For example, if the following routes are summarized:

```
OI  2001:DB8:0:7::/64 [110/20]
     via FE80::A8BB:CCFF:FE00:6F00, GigabitEthernet0/0/0
OI  2001:DB8:0:8::/64 [110/100]
     via FE80::A8BB:CCFF:FE00:6F00, GigabitEthernet0/0/0
OI  2001:DB8:0:9::/64 [110/20]
     via FE80::A8BB:CCFF:FE00:6F00, GigabitEthernet0/0/0
```

They become one summarized route, as follows:

```
OI  2001:DB8::/48 [110/100]
     via FE80::A8BB:CCFF:FE00:6F00, GigabitEthernet0/0/0
```

OSPFv3 routing must be enabled.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **router ospfv3** [*process-id*]
4. **address-family ipv6 unicast**
5. **area** *area-ID* **range** *ipv6-prefix*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**router ospfv3** [*process-id*] **Example:** `Router(config)# router ospfv3 1`	Enables OSPFv3 router configuration mode for the IPv4 or IPv6 address family.

Command or Action	Purpose

Step 4 **address-family ipv6 unicast**
 	
Example:
 	
Example:
or
 	
Example:
address-family ipv4
unicast
 	
Example:
`Router(config-router)# address-family ipv6 unicast`
 	
Example:
 	
Example:
or
 	
Example:
`Router(config-router)# address-family ipv4 unicast`
	Enters IPv6 address family configuration mode for OSPFv3.
	or
	Enters IPv4 address family configuration mode for OSPFv3.

Step 5 **area** *area-ID* **range** *ipv6-prefix* **Example:** `Router(config-router-af)# area 1 range 2001:DB8:0:0::0/128`	Configures OSPFv3 area parameters.

- Defining an OSPFv3 Area Range, page 498

Defining an OSPFv3 Area Range

This task can be performed in releases prior to Cisco IOS XE Release 3.4S.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ipv6 router ospf** *process-id*
4. **area** *area-id* **range** *ipv6-prefix* / *prefix-length* **advertise** | **not-advertise**] [**cost** *cost*]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**ipv6 router ospf** *process-id* **Example:** `Router(config)# ipv6 router ospf 1`	Enables OSPFv3 router configuration mode.
Step 4	**area** *area-id* **range** *ipv6-prefix* / *prefix-length* **advertise**	**not-advertise**] [**cost** *cost*] **Example:** `Router(config-rtr)# area 1 range 2001:DB8::/48`	Consolidates and summarizes routes at an area boundary.

Configuring the OSPFv3 Max-Metric Router LSA

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **router ospfv3** *process-id*
4. **max-metric router-lsa** [**external-lsa** [*max-metric-value*]] [**include-stub**] [**inter-area-lsas** [*max-metric-value*]] [**on-startup** {*seconds* | **wait-for-bgp**}] [**prefix-lsa**] [**stub-prefix-lsa** [*max-metric-value*]] [**summary-lsa** [*max-metric-value*]]
5. **exit**
6. **show ospfv3** [*process-id*] **max-metric**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**router ospfv3** *process-id* **Example:** `Device(config)# router ospfv3 1`	Enables OSPFv3 router configuration mode.
Step 4	**max-metric router-lsa** [**external-lsa** [*max-metric-value*]] [**include-stub**] [**inter-area-lsas** [*max-metric-value*]] [**on-startup** {*seconds* | **wait-for-bgp**}] [**prefix-lsa**] [**stub-prefix-lsa** [*max-metric-value*]] [**summary-lsa** [*max-metric-value*]] **Example:** `Device(config-router)# max-metric router-lsa on-startup wait-for-bgp`	Configures a device that is running the OSPFv3 protocol to advertise a maximum metric so that other devices do not prefer the device as an intermediate hop in their SPF calculations.

	Command or Action	Purpose
Step 5	**exit** **Example:** `Device(config-router)# exit`	Leaves the current configuration mode. • Enter this command twice to reach privileged EXEC mode.
Step 6	**show ospfv3** [*process-id*] **max-metric** **Example:** `Device# show ospfv3 1 max-metric`	Displays OSPFv3 maximum metric origination information.

Configuring IPsec on OSPFv3

Once you have configured OSPFv3 and decided on your authentication, you must define the security policy on each of the routers within the group. The security policy consists of the combination of the key and the SPI. To define a security policy, you must define an SPI and a key.

You can configure an authentication or encryption policy either on an interface or for an OSPFv3 area. When you configure for an area, the security policy is applied to all of the interfaces in the area. For higher security, use a different policy on each interface.

You can configure authentication and encryption on virtual links.

- Defining Authentication on an Interface, page 501
- Defining Encryption on an Interface, page 502
- Defining Authentication in an OSPFv3 Area, page 504
- Defining Encryption in an OSPFv3 Area, page 505

Defining Authentication on an Interface

Before you configure IPsec on an interface, you must configure OSPFv3 on that interface.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type number*
4. Do one of the following:
   - **ospfv3 authentication** {**ipsec** *spi*} {**md5** | **sha1**} *key-encryption-type key*} | **null**
   - **ipv6 ospf authentication ipsec spi** *spi* **md5** *key-encryption-type* {*key* | **null**}]

DETAILED STEPS

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

Step 4	Do one of the following:
	• **ospfv3 authentication** {**ipsec** *spi*} {**md5** | **sha1**} *key-encryption-type key*} | **null**
	• **ipv6 ospf authentication ipsec spi** *spi* **md5** *key-encryption-type* {*key* | **null**}]
	
	Example:
	`Router(config-if)# ospfv3 authentication md5 0 27576134094768132473302031209727`
	
	Example:
	`Router(config-if)# ipv6 ospf authentication ipsec spi 500 md5 1234567890abcdef1234567890abcdef`
		Specifies the authentication type for an interface.

Defining Encryption on an Interface

Before you configure IPsec on an interface, you must configure OSPFv3 on that interface.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type number*
4. Do one of the following:
   - **ospfv3 encryption** {**ipsec spi** *spi* **esp** *encryption-algorithm* {*key-encryption-type key*} *authentication-algorithm* {*key-encryption-type key*} | **null**}
   -
   - **ipv6 ospf encryption ipsec spi** *spi* **esp** *encryption-algorithm* [[*key-encryption-type*] *key*] *authentication-algorithm key-encryption-type*] *key* | **null**

DETAILED STEPS

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

Command or Action	Purpose

Step 4 Do one of the following:
• **ospfv3 encryption** {**ipsec spi** *spi* **esp** *encryption-algorithm* {*key-encryption-type key*} *authentication-algorithm* {*key-encryption-type key*} | **null**}
•
• **ipv6 ospf encryption ipsec spi** *spi* **esp** *encryption-algorithm* [[*key-encryption-type*] *key*] *authentication-algorithm key-encryption-type*] *key* | **null**
Example:
`Router(config-if)# ospfv3 encryption ipsec spi 1001 esp null md5 0 275761340947681324733020312097 27`
Example:
`Router(config-if) ipv6 ospf encryption ipsec spi 1001 esp null sha1 123456789A123456789B123456789C123456789D`
	Specifies the encryption type for an interface.

Defining Authentication in an OSPFv3 Area

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ipv6 router ospf** *process-id*
4. **area** *area-id* **authentication ipsec spi** *spi authentication-algorithm* [*key-encryption-type*] *key*

DETAILED STEPS

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

	Command or Action	Purpose
Step 3	**ipv6 router ospf** *process-id* **Example:** `Device(config)# ipv6 router ospf 1`	Enables OSPFv3 router configuration mode.
Step 4	**area** *area-id* **authentication ipsec spi** *spi authentication-algorithm* [*key-encryption-type*] *key* **Example:** `Device(config-rtr)# area 1 authentication ipsec spi 678 md5 1234567890ABCDEF1234567890ABCDEF`	Enables authentication in an OSPFv3 area.

Defining Encryption in an OSPFv3 Area

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ipv6 router ospf** *process-id*
4. **area** *area-id* **encryption ipsec spi** *spi* **esp** { *encryption-algorithm* [| *key-encryption-type*] *key* | **null**} *authentication-algorithm* [| *key-encryption-type*] *key*

DETAILED STEPS

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

	Command or Action	Purpose
Step 3	**ipv6 router ospf** *process-id* **Example:** `Device(config)# ipv6 router ospf 1`	Enables OSPFv3 router configuration mode.
Step 4	**area** *area-id* **encryption ipsec spi** *spi* **esp** { *encryption-algorithm* [| *key-encryption-type*] *key* | **null**} *authentication-algorithm* [| *key-encryption-type*] *key* **Example:** `Device(config-rtr)# area 1 encryption ipsec spi 500 esp null md5 1aaa2bbb3ccc4ddd5eee6fff7aaa8bbb`	Enables encryption in an OSPFv3 area.

Tuning LSA and SPF Transmission for OSPFv3 Fast Convergence

The task can be performed in Cisco IOS XE Release 3.4S and later releases.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **router ospfv3** [*process-id*]
4. **timers lsa arrival** *milliseconds*
5. **timers pacing flood** *milliseconds*
6. **timers pacing lsa-group** *seconds*
7. **timers pacing retransmission** *milliseconds*

DETAILED STEPS

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

	Command or Action	Purpose
Step 3	**router ospfv3** [*process-id*] **Example:** `Router(config)# router ospfv3 1`	Enables OSPFv3 router configuration mode for the IPv4 or IPv6 address family.
Step 4	**timers lsa arrival** *milliseconds* **Example:** `Router(config-rtr)# timers lsa arrival 300`	Sets the minimum interval at which the software accepts the same LSA from OSPFv3 neighbors.
Step 5	**timers pacing flood** *milliseconds* **Example:** `Router(config-rtr)# timers pacing flood 30`	Configures LSA flood packet pacing.
Step 6	**timers pacing lsa-group** *seconds* **Example:** `Router(config-router)# timers pacing lsa-group 300`	Changes the interval at which OSPFv3 LSAs are collected into a group and refreshed, checksummed, or aged.
Step 7	**timers pacing retransmission** *milliseconds* **Example:** `Router(config-router)# timers pacing retransmission 100`	Configures LSA retransmission packet pacing in IPv4 OSPFv3.

Configuring LSA and SPF Throttling for OSPFv3 Fast Convergence

The task can be performed in releases prior to Cisco IOS XE Release 3.4S.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ipv6 router ospf** *process-id*
4. **timers throttle spf** *spf-start spf-hold spf-max-wait*
5. **timers throttle lsa** *start-interval hold-interval max-interval*
6. **timers lsa arrival** *milliseconds*
7. **timers pacing flood** *milliseconds*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**ipv6 router ospf** *process-id* **Example:** `Router(config)# ipv6 router ospf 1`	Enables OSPFv3 router configuration mode.
Step 4	**timers throttle spf** *spf-start spf-hold spf-max-wait* **Example:** `Router(config-rtr)# timers throttle spf 200 200 200`	Turns on SPF throttling.
Step 5	**timers throttle lsa** *start-interval hold-interval max-interval* **Example:** `Router(config-rtr)# timers throttle lsa 300 300 300`	Sets rate-limiting values for OSPFv3 LSA generation.
Step 6	**timers lsa arrival** *milliseconds* **Example:** `Router(config-rtr)# timers lsa arrival 300`	Sets the minimum interval at which the software accepts the same LSA from OSPFv3 neighbors.
Step 7	**timers pacing flood** *milliseconds* **Example:** `Router(config-rtr)# timers pacing flood 30`	Configures LSA flood packet pacing.

Enabling Event Logging for LSA and SPF Rate Limiting for the IPv6 or IPv4 Address Family

This task can be performed in Cisco IOS XE Release 3.4S and later releases.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **router ospfv3** [*process-id*]
4. **address-family ipv6 unicast**
5. **event-log** [**one-shot** | **pause** | **size** *number-of-events*]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**router ospfv3** [*process-id*] **Example:** `Router(config)# router ospfv3`	Enables OSPFv3 router configuration mode for the IPv4 or IPv6 address family.

Command or Action	Purpose

Step 4 **address-family ipv6 unicast**
 	
Example:
 	
Example:
or
 	
Example:
address-family ipv4
unicast
 	
Example:
`Router(config-router)# address-family ipv6 unicast`
 	
Example:
 	
Example:
or
 	
Example:
`Router(config-router)# address-family ipv4 unicast`
	Enters IPv6 address family configuration mode for OSPFv3.
	or
	Enters IPv4 address family configuration mode for OSPFv3.

Step 5 **event-log** [**one-shot** | **pause** | **size** *number-of-events*] **Example:** `Router(config-router)# event-log`	Enable OSPFv3 event logging in an IPv4 OSPFv3 process.

- Enabling Event Logging for LSA and SPF Rate Limiting, page 510
- Clearing the Content of an Event Log, page 511

Enabling Event Logging for LSA and SPF Rate Limiting

This task can be performed in releases prior to Cisco IOS XE Release 3.4S.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ipv6 router ospf** *process-id*
4. **event-log** [**size** [*number of events*]] [**one-shot**] [**pause**]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**ipv6 router ospf** *process-id* **Example:** `Router(config)# ipv6 router ospf 1`	Enables OSPFv3 router configuration mode.
Step 4	**event-log** [**size** [*number of events*]] [**one-shot**] [**pause**] **Example:** `Router(config-rtr)# event-log size 10000 one-shot`	Enables event logging.

Clearing the Content of an Event Log

SUMMARY STEPS

1. **enable**
2. **clear ipv6 ospf** [*process-id*] **events**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**clear ipv6 ospf** [*process-id*] **events** **Example:** `Router# clear ipv6 ospf 1 events`	Clears the OSPFv3 event log content based on the OSPFv3 routing process ID.

Calculating OSPFv3 External Path Preferences per RFC 5340

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **router ospfv3** [*process-id*]
4. **no compatible rfc1583**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**router ospfv3** [*process-id*] **Example:** `Device(config)# router ospfv3 1`	Enables OSPFv3 router configuration mode for the IPv4 or IPv6 address family.

	Command or Action	Purpose
Step 4	**no compatible rfc1583** **Example:** `Device(config-router)# no compatible rfc1583`	Changes the method used to calculate external path preferences per RFC 5340.

Enabling OSPFv3 Graceful Restart

Enabling OSPFv3 Graceful Restart on a Graceful-Restart-Capable Router

The task can be performed in Cisco IOS XE 3.4S and later releases.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **router ospfv3** [*process-id*]
4. **graceful-restart** [**restart-interval** *interval*]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**router ospfv3** [*process-id*] **Example:** `Router(config)# router ospfv3 1`	Enables OSPFv3 router configuration mode for the IPv4 or IPv6 address family.

	Command or Action	Purpose
Step 4	**graceful-restart** [**restart-interval** *interval*] **Example:** `Router(config-rtr)# graceful-restart`	Enables the OSPFv3 graceful restart feature on a graceful-restart-capable router.

- Enabling OSPFv3 Graceful Restart on a Graceful-Restart-Capable Router, page 514

Enabling OSPFv3 Graceful Restart on a Graceful-Restart-Capable Router

The task can be performed in releases prior to Cisco IOS XE Release 3.4S.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ipv6 router ospf** *process-id*
4. **graceful-restart** [**restart-interval** *interval*]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**ipv6 router ospf** *process-id* **Example:** `Router(config)# ipv6 router ospf 1`	Enables OSPFv3 router configuration mode.
Step 4	**graceful-restart** [**restart-interval** *interval*] **Example:** `Router(config-rtr)# graceful-restart`	Enables the OSPFv3 graceful restart feature on a graceful-restart-capable router.

Enabling OSPFv3 Graceful Restart on a Graceful-Restart-Aware Router

The task can be performed in Cisco IOS XE Release 3.4S and later releases.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **router ospfv3** [*process-id*]
4. **graceful-restart helper** {**disable** | **strict-lsa-checking**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**router ospfv3** [*process-id*] **Example:** `Router(config)# router ospfv3 1`	Enables OSPFv3 router configuration mode for the IPv4 or IPv6 address family.
Step 4	**graceful-restart helper** {**disable**	**strict-lsa-checking** **Example:** `Router(config-rtr)# graceful-restart helper strict-lsa-checking`	Enables the OSPFv3 graceful restart feature on a graceful-restart-aware router.

Example:

- Enabling OSPFv3 Graceful Restart on a Graceful-Restart-Aware Router, page 515

Enabling OSPFv3 Graceful Restart on a Graceful-Restart-Aware Router

The task can be performed in releases prior to Cisco IOS XE Release 3.4S.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ipv6 router ospf** *process-id*
4. **graceful-restart helper** {**disable** | **strict-lsa-checking**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**ipv6 router ospf** *process-id* **Example:** `Router(config)# ipv6 router ospf 1`	Enables OSPFv3 router configuration mode.
Step 4	**graceful-restart helper** {**disable**	**strict-lsa-checking** **Example:** `Router(config-rtr)# graceful-restart helper strict-lsa-checking`	Enables the OSPFv3 graceful restart feature on a graceful-restart-aware router.

Example:

Forcing an SPF Calculation

SUMMARY STEPS

1. **enable**
2. **clear ospfv3** [*process-id*] **force-spf**
3. **clear ospfv3** [*process-id*] **process**
4. **clear ospfv3** [*process-id*] **redistribution**
5. **clear ipv6 ospf** [*process-id*] {**process** | **force-spf** | **redistribution**}

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**clear ospfv3** [*process-id*] **force-spf** **Example:** `Device# clear ospfv3 1 force-spf`	Runs SPF calculations for an OSPFv3 process. • If the **clear ospfv3 force-spf** command is configured, it overwrites the **clear ipv6 ospf** configuration. • Once the **clear ospfv3 force-spf** command has been used, the **clear ipv6 ospf** command cannot be used.
Step 3	**clear ospfv3** [*process-id*] **process** **Example:** `Device# clear ospfv3 2 process`	Resets an OSPFv3 process. • If the **clear ospfv3 force-spf** command is configured, it overwrites the **clear ipv6 ospf** configuration. • Once the **clear ospfv3 force-spf** command has been used, the **clear ipv6 ospf** command cannot be used.
Step 4	**clear ospfv3** [*process-id*] **redistribution** **Example:** `Device# clear ospfv3 redistribution`	Clears OSPFv3 route redistribution. • If the **clear ospfv3 force-spf** command is configured, it overwrites the **clear ipv6 ospf** configuration. • Once the **clear ospfv3 force-spf** command has been used, the **clear ipv6 ospf** command cannot be used.
Step 5	**clear ipv6 ospf** [*process-id*] {**process**	**force-spf**	**redistribution**} **Example:** `Device# clear ipv6 ospf force-spf`	Clears the OSPFv3 state based on the OSPFv3 routing process ID, and forces the start of the SPF algorithm. • If the **clear ospfv3 force-spf** command is configured, it overwrites the **clear ipv6 ospf** configuration. • Once the **clear ospfv3 force-spf** command has been used, the **clear ipv6 ospf** command cannot be used.

Verifying OSPFv3 Configuration and Operation

This task is optional. The commands in this task are available in Cisco IOS XE Release 3.4S and later releases.

SUMMARY STEPS

1. **enable**
2. **show ospfv3** [*process-id*] **border-routers**
3. **show ospfv3** [*process-id* [*area-id*]] **database** [**database-summary** | **internal** | **external**[*ipv6-prefix*] [*link-state-id*] | **grace** | **inter-area prefix** [*ipv6-prefix* | *link-state-id*] | **inter-area router** [*destination-router-id* | *link-state-id*] | **link** [**interface** *interface-name* | *link-state-id*] | **network** [*link-state-id*] | **nssa-external** [*ipv6-prefix*] [*link-state-id*] | **prefix** [**ref-lsa** {**router** | **network**} | *link-state-id*] | **promiscuous** | **router** [*link-state-id*] | **unknown** [{**a rea** | **as** | **link**} [*link-state-id*]] [**adv-router** *router-id*] [**self-originate**]
4. **show ospfv3** [*process-id*] **events** [**generic** | **interface** | **lsa** | **neighbor** | **reverse** | **rib** | **spf**]
5. show ospfv3 [*process-id*] [*area-id*] flood-list interface-*type interface-number*
6. show ospfv3 [*process-id*] graceful-restart
7. **show ospfv3** [*process-id*] [*area-id*] **interface**[*type number*] [**brief**]
8. **show ospfv3** [*process-id*] [*area-id*] **neighbor**[*interface type interface-number*] [*neighbor-id*] [**detail**]
9. **show ospfv3** [*process-id*] [*area-id*] **request-list**[*neighbor*] [*interface*] [*interface neighbor*]
10. **show ospfv3** [*process-id*] [*area-id*] **retransmission-list** [*neighbor*] [*interface*] [*interface neighbor*]
11. **show ospfv3** [*process-id*] **statistic**[**detail**]
12. **show ospfv3** [*process-id*] **summary-prefix**
13. **show ospfv3** [*process-id*] **timers rate-limit**
14. **show ospfv3** [*process-id*] **traffic**[*interface-type interface-number*]
15. **show ospfv3** [*process-id*] **virtual-links**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**show ospfv3** [*process-id*] **border-routers** **Example:** `Router# show ospfv3 border-routers`	Displays the internal OSPFv3 routing table entries to an ABR and ASBR.

	Command or Action	Purpose
Step 3	**show ospfv3** [*process-id* [*area-id*]] **database** [**database-summary** | **internal** | **external**[*ipv6-prefix*] [*link-state-id*] | **grace** | **inter-area prefix** [*ipv6-prefix* | *link-state-id*] | **inter-area router** [*destination-router-id* | *link-state-id*] | **link** [**interface** *interface-name* | *link-state-id*] | **network** [*link-state-id*] | **nssa-external** [*ipv6-prefix*] [*link-state-id*] | **prefix** [**ref-lsa** {**router** | **network**} | *link-state-id*] | **promiscuous** | **router** [*link-state-id*] | **unknown** [{**a rea** | **as** | **link**} [*link-state-id*]] [**adv-router** *router-id*] [**self-originate**] **Example:** `Router# show ospfv3 database`	Displays lists of information related to the OSPFv3 database for a specific router.
Step 4	**show ospfv3** [*process-id*] **events** [**generic** | **interface** | **lsa** | **neighbor** | **reverse** | **rib** | **spf**] **Example:** `Router# show ospfv3 events`	Displays detailed information about OSPFv3 events.
Step 5	show ospfv3 [*process-id*] [*area-id*] flood-list interface-*type interface-number* **Example:** `Router# show ospfv3 flood-list`	Displays a list of OSPFv3 LSAs waiting to be flooded over an interface.
Step 6	show ospfv3 [*process-id*] graceful-restart **Example:** `Router# show ospfv3 graceful-restart`	Displays OSPFv3 graceful restart information.
Step 7	**show ospfv3** [*process-id*] [*area-id*] **interface**[*type number*] [**brief**] **Example:** `Router# show ospfv3 interface`	Displays OSPFv3-related interface information.
Step 8	**show ospfv3** [*process-id*] [*area-id*] **neighbor**[*interface type interface-number*] [*neighbor-id*] [**detail**] **Example:** `Router# show ospfv3 neighbor`	Displays OSPFv3 neighbor information on a per-interface basis.

	Command or Action	Purpose
Step 9	**show ospfv3** [*process-id*] [*area-id*] **request-list**[*neighbor*] [*interface*] [*interface neighbor*] **Example:** `Router# show ospfv3 request-list`	Displays a list of all LSAs requested by a router.
Step 10	**show ospfv3** [*process-id*] [*area-id*] **retransmission-list** [*neighbor*] [*interface*] [*interface neighbor*] **Example:** `Router# show ospfv3 retransmission-list`	Displays a list of all LSAs waiting to be re-sent.
Step 11	**show ospfv3** [*process-id*] **statistic**[**detail**] **Example:** `Router# show ospfv3 statistics`	Displays OSPFv3 SPF calculation statistics.
Step 12	**show ospfv3** [*process-id*] **summary-prefix** **Example:** `Router# show ospfv3 summary-prefix`	Displays a list of all summary address redistribution information configured under an OSPFv3 process.
Step 13	**show ospfv3** [*process-id*] **timers rate-limit** **Example:** `Router# show ospfv3 timers rate-limit`	Displays all of the LSAs in the rate limit queue.
Step 14	**show ospfv3** [*process-id*] **traffic**[*interface-type interface-number*] **Example:** `Router# show ospfv3 traffic`	Displays OSPFv3 traffic statistics.
Step 15	**show ospfv3** [*process-id*] **virtual-links** **Example:** `Router# show ospfv3 virtual-links`	Displays parameters and the current state of OSPFv3 virtual links.

- Verifying OSPFv3 Configuration and Operation, page 521
- Examples, page 521

Verifying OSPFv3 Configuration and Operation

SUMMARY STEPS

1. **enable**
2. **show ipv6 ospf** [*process-id*] [*area-id*] **interface**[*interface-type interface-number*]
3. **show ipv6 ospf** [*process-id*] [*area-id*]
4. **show ipv6 ospf** [*process-ID*] **event** [**generic** | **interface** | **lsa** | **neighbor** | **reverse** | **rib** | **spf**]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**show ipv6 ospf** [*process-id*] [*area-id*] **interface**[*interface-type interface-number*] **Example:** `Router# show ipv6 ospf interface`	Displays OSPFv3-related interface information.
Step 3	**show ipv6 ospf** [*process-id*] [*area-id*] **Example:** `Router# show ipv6 ospf`	Displays general information about OSPFv3 routing processes.
Step 4	**show ipv6 ospf** [*process-ID*] **event** [**generic** | **interface** | **lsa** | **neighbor** | **reverse** | **rib** | **spf**] **Example:** `Router# show ipv6 ospf event spf`	Displays detailed information about OSPFv3 events.

Examples

- Sample Output for the show ipv6 ospf interface Command, page 521
- Sample Output for the show ipv6 ospf Command, page 523
- Sample Output for the show ipv6 ospf graceful-restart Command, page 523

Sample Output for the show ipv6 ospf interface Command

The following is sample output from the **show ipv6 ospf interface** command with regular interfaces and a virtual link that are protected by encryption and authentication:

```
Router# show ipv6 ospf interface
OSPFv3_VL1 is up, line protocol is up
   Interface ID 69
   Area 0, Process ID 1, Instance ID 0, Router ID 10.0.0.1
   Network Type VIRTUAL_LINK, Cost: 64
   Configured as demand circuit.
   Run as demand circuit.
   DoNotAge LSA allowed.
   NULL encryption SHA-1 auth SPI 3944, secure socket UP (errors: 0)
   Transmit Delay is 1 sec, State POINT_TO_POINT,
   Timer intervals configured, Hello 2, Dead 10, Wait 40, Retransmit 5
     Hello due in 00:00:00
   Index 1/3/5, flood queue length 0
   Next 0x0(0)/0x0(0)/0x0(0)
   Last flood scan length is 1, maximum is 1
   Last flood scan time is 0 msec, maximum is 0 msec
   Neighbor Count is 1, Adjacent neighbor count is 1
     Adjacent with neighbor 10.2.0.1  (Hello suppressed)
   Suppress hello for 1 neighbor(s)
OSPFv3_VL0 is up, line protocol is up
   Interface ID 67
   Area 0, Process ID 1, Instance ID 0, Router ID 10.0.0.1
   Network Type VIRTUAL_LINK, Cost: 128
   Configured as demand circuit.
   Run as demand circuit.
   DoNotAge LSA allowed.
   MD5 authentication SPI 940, secure socket UP (errors: 0)
   Transmit Delay is 1 sec, State POINT_TO_POINT,
   Timer intervals configured, Hello 10, Dead 40, Wait 40, Retransmit 5
     Hello due in 00:00:09
   Index 1/2/4, flood queue length 0
   Next 0x0(0)/0x0(0)/0x0(0)
   Last flood scan length is 1, maximum is 10
   Last flood scan time is 0 msec, maximum is 0 msec
   Neighbor Count is 1, Adjacent neighbor count is 1
     Adjacent with neighbor 10.1.0.1  (Hello suppressed)
   Suppress hello for 1 neighbor(s)
Gigabitethernet1/0/0 is up, line protocol is up
   Link Local Address FE80::A8BB:CCFF:FE00:6601, Interface ID 6
   Area 0, Process ID 1, Instance ID 0, Router ID 10.0.0.1
   Network Type BROADCAST, Cost: 10
   Transmit Delay is 1 sec, State DR, Priority 1
   Designated Router (ID) 10.0.0.1, local address FE80::A8BB:CCFF:FE00:6601
   No backup designated router on this network
   Timer intervals configured, Hello 10, Dead 40, Wait 40, Retransmit 5
     Hello due in 00:00:09
   Index 1/1/1, flood queue length 0
   Next 0x0(0)/0x0(0)/0x0(0)
   Last flood scan length is 0, maximum is 0
   Last flood scan time is 0 msec, maximum is 0 msec
   Neighbor Count is 0, Adjacent neighbor count is 0
   Suppress hello for 0 neighbor(s)
Serial12/0 is up, line protocol is up
   Link Local Address FE80::A8BB:CCFF:FE00:6600, Interface ID 50
   Area 1, Process ID 1, Instance ID 0, Router ID 10.0.0.1
   Network Type POINT_TO_POINT, Cost: 64
   AES-CBC encryption SHA-1 auth SPI 2503, secure socket UP (errors: 0)
   authentication NULL
   Transmit Delay is 1 sec, State POINT_TO_POINT,
   Timer intervals configured, Hello 10, Dead 40, Wait 40, Retransmit 5
     Hello due in 00:00:09
   Index 1/2/3, flood queue length 0
   Next 0x0(0)/0x0(0)/0x0(0)
   Last flood scan length is 1, maximum is 5
   Last flood scan time is 0 msec, maximum is 0 msec
   Neighbor Count is 1, Adjacent neighbor count is 1
     Adjacent with neighbor 10.2.0.1
   Suppress hello for 0 neighbor(s)
Serial11/0 is up, line protocol is up
```

```
   Link Local Address FE80::A8BB:CCFF:FE00:6600, Interface ID 46
   Area 1, Process ID 1, Instance ID 0, Router ID 10.0.0.1
   Network Type POINT_TO_POINT, Cost: 64
   MD5 authentication (Area) SPI 500, secure socket UP (errors: 0)
   Transmit Delay is 1 sec, State POINT_TO_POINT,
   Timer intervals configured, Hello 10, Dead 40, Wait 40, Retransmit 5
     Hello due in 00:00:09
   Index 1/1/2, flood queue length 0
   Next 0x0(0)/0x0(0)/0x0(0)
   Last flood scan length is 1, maximum is 5
   Last flood scan time is 0 msec, maximum is 0 msec
   Neighbor Count is 1, Adjacent neighbor count is 1
     Adjacent with neighbor 1.0.0.1
   Suppress hello for 0 neighbor(s)
```

Sample Output for the show ipv6 ospf Command

The following is sample output from the **show ipv6 ospf** command:

```
Router# show ipv6 ospf
Routing Process "ospfv3 1" with ID 172.16.3.3
 It is an autonomous system boundary router
 Redistributing External Routes from,
    static
 SPF schedule delay 5 secs, Hold time between two SPFs 10 secs
 Minimum LSA interval 5 secs. Minimum LSA arrival 1 secs
 LSA group pacing timer 240 secs
 Interface flood pacing timer 33 msecs
 Retransmission pacing timer 66 msecs
 Number of external LSA 1. Checksum Sum 0x218D
 Number of areas in this router is 1. 1 normal 0 stub 0 nssa
    Area 1
        Number of interfaces in this area is 2
        SPF algorithm executed 9 times
        Number of LSA 15. Checksum Sum 0x67581
        Number of DCbitless LSA 0
        Number of indication LSA 0
        Number of DoNotAge LSA 0
        Flood list length 0
```

Sample Output for the show ipv6 ospf graceful-restart Command

The following is sample output from the **show ipv6 ospf graceful-restart**command:

```
Router# show ipv6 ospf graceful-restart
Routing Process "ospf 1"
 Graceful Restart enabled
    restart-interval limit: 120 sec, last restart 00:00:15 ago (took 36 secs)
  Graceful Restart helper support enabled
  Router status : Active
  Router is running in SSO mode
  OSPF restart state : NO_RESTART
  Router ID 10.1.1.1, checkpoint Router ID 10.0.0.0
```

Configuration Examples for Implementing OSPFv3

- Example: Enabling OSPFv3 on an Interface Configuration, page 524
- Example: Defining an OSPFv3 Area Range, page 524
- Example: Configuring LSA and SPF Throttling for OSPFv3 Fast Convergence, page 524
- Example: Forcing SPF Configuration, page 524

Example: Enabling OSPFv3 on an Interface Configuration

The following example shows the command to use to configure OSPFv3 routing process 109 to run on the interface and puts it in area 1:

```
ipv6 ospf 109 area 1
```

Example: Defining an OSPFv3 Area Range

```
interface gigabitethernet7/0/0
 ipv6 address 2001:DB8:0:7::/64 eui-64
 ipv6 enable
 ipv6 ospf 1 area 1
!
interface gigabitethernet8/0/0
 ipv6 address 2001:DB8:0:8::/64 eui-64
 ipv6 enable
 ipv6 ospf 1 area 1
!
interface gigabitethernet9/0/0
 ipv6 address 2001:DB8:0:9::/64 eui-64
 ipv6 enable
 ipv6 ospf 1 area 1
!
ipv6 router ospf 1
 router-id 10.11.11.1
 area 1 range 2001:DB8::/48
```

Example: Configuring LSA and SPF Throttling for OSPFv3 Fast Convergence

The following example show how to display the configuration values for SPF and LSA throttling timers:

```
Router# show ipv6 ospf

Routing Process "ospfv3 1" with ID 10.9.4.1
Event-log enabled, Maximum number of events: 1000, Mode: cyclic
 It is an autonomous system boundary router
 Redistributing External Routes from,
    ospf 2
 Initial SPF schedule delay 5000 msecs
 Minimum hold time between two consecutive SPFs 10000 msecs
 Maximum wait time between two consecutive SPFs 10000 msecs
 Minimum LSA interval 5 secs
 Minimum LSA arrival 1000 msecs
```

Example: Forcing SPF Configuration

The following example shows how to trigger SPF to redo the SPF and repopulate the routing tables:

```
clear ipv6 ospf force-spf
```

Additional References

Related Documents

Related Topic	Document Title
Configuring a router ID in OSPF	• " Configuring OSPF ," *Cisco IOS XE IP Routing Protocols Configuration Guide* • *Cisco IOS IP Routing Protocols Command Reference*
LSA throttling	"OSPF Link-State Advertisement (LSA) Throttling ," *Cisco IOS XE IP Routing Protocols Configuration Guide*
OSPFv3 commands	*Cisco IOS IPv6 Command Reference*
IPv6 supported feature list	" Start Here: Cisco IOS XE Software Release Specifics for IPv6 Features ," *Cisco IOS XE IPv6 Configuration Guide*
Implementing basic IPv6 connectivity	" Implementing IPv6 Addressing and Basic Connectivity ," *Cisco IOS XE IPv6 Configuration Guide*
Stateful switchover	"Stateful Switchover ," *Cisco IOS XE High Availability Configuration Guide*
Cisco nonstop forwarding	"Cisco Nonstop Forwarding ," *Cisco IOS XE High Availability Configuration Guide*
OSPF for IPv4 commands	*Cisco IOS IP Routing Protocols Command Reference*
Security configuration tasks (IPv4)	*Cisco IOS XE Security Configuration Guide* , Release 2
Security commands: complete command syntax, command mode, defaults, usage guidelines, and examples (IPv4)	*Cisco IOS Security Command Reference*
Cisco IOS master command list, all releases	Cisco IOS Master Command List, All Releases

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
None	To locate and download MIBs for selected platforms, Cisco software releases, and feature sets, use Cisco MIB Locator found at the following URL: http://www.cisco.com/go/mibs

RFCs

RFCs	Title
RFC 1583	*OSPF version 2*
RFC 2401	*Security Architecture for the Internet Protocol*
RFC 2402	*IP Authentication Header*
RFC 2406	*IP Encapsulating Security Payload (ESP)*
RFC 3137	OSPF Stub Router Advertisement
RFC 4552	*Authentication/Confidentiality for OSPFv3*
RFC 5187	*OSPFv3 Graceful Restart*
RFC 5340	*OSPF for IPv6*
RFC 5838	*Support of Address Families in OSPFv3*

Technical Assistance

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

Feature Information for Implementing OSPFv3

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Table 23 Feature Information for Implementing OSPFv3

Feature Name	Releases	Feature Information
IPv6 Routing--Fast Convergence--LSA and SPF Throttling	Cisco IOS XE Release 2.1	The OSPFv3 LSA and SPF throttling feature provides a dynamic mechanism to slow down link-state advertisement updates in OSPFv3 during times of network instability. The following commands were modified by this feature: **clear ipv6 ospf events**, **event-log**, **ipv6 router ospf**, **show ipv6 ospf event**, **timers lsa arrival**, **timers pacing flood**, **timers throttle lsa**, **timers throttle spf**
IPv6 Routing--LSA Types in OSPFv3	Cisco IOS XE Release 2.1	A router's collection of LSA data is stored in a link-state database. The contents of the database, when subjected to the Dijkstra algorithm, result in the creation of the OSPFv3 routing table.
IPv6 Routing-- OSPFv3	Cisco IOS XE Release 2.1	OSPF version 3 for IPv6 expands on OSPF version 2 to provide support for IPv6 routing prefixes and the larger size of IPv6 addresses. The following commands were modified by this feature: **area range**, **clear ipv6 ospf**, **ipv6 ospf area**, **ipv6 router ospf**, **show ipv6 ospf**, **show ipv6 ospf interface**
OSPFv3 Address Families	Cisco IOS XE Release 3.4S	The OSPFv3 address families feature enables IPv4 and IPv6 unicast traffic to be supported with a single network topology.
OSPFv3 External Path Preference Option	Cisco IOS XE Release 3.4S	This feature is provides a way to calculate external path preferences per RFC 5340.

Feature Name	Releases	Feature Information
OSPFv3 Graceful Restart	Cisco IOS XE Release 2.1 Cisco IOS XE Release 3.3SG	The graceful restart feature in OSPFv3 allows nonstop data forwarding along routes that are already known while the OSPFv3 routing protocol information is being restored. The following commands were modified by this feature: **graceful-restart**, **graceful-restart helper**, **ipv6 router ospf**, **show ipv6 ospf graceful-restart**
OSPFv3 Max-Metric Router LSA	Cisco IOS XE Release 3.4S	The OSPFv3 max-mtric router LSA feature enables OSPF to advertise its locally generated router LSAs with a maximum metric.
OSPFv3 IPSec ESP Encryption and Authentication	Cisco IOS XE Release 3.3SG	Supports ESP authentication and encryption, including virtual links.

Implementing Policy-Based Routing for IPv6

This module describes policy-based routing (PBR) for IPv6. PBR in both IPv6 and IPv4 allows a user to manually configure how received packets should be routed. PBR allows the user to identify packets using several attributes and to specify the next hop or output interface to which the packet should be sent. PBR also provides a basic packet-marking capability.

Finding Feature Information

Your software release may not support all the features documented in this module. For the latest caveats and feature information, see Bug Search Tool and the release notes for your platform and software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the feature information table at the end of this module.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Information About Implementing Policy-Based Routing for IPv6

Policy-Based Routing Overview

PBR gives you a flexible means of routing packets by allowing you to configure a defined policy for traffic flows, which lessens reliance on routes derived from routing protocols. To this end, PBR gives you more control over routing by extending and complementing the existing mechanisms provided by routing

protocols. PBR allows you to set the IPv6 precedence. It also allows you to specify a path for certain traffic, such as priority traffic over a high-cost link.

PBR for IPv6 may be applied to both forwarded and originated IPv6 packets. For forwarded packets, PBR for IPv6 will be implemented as an IPv6 input interface feature, supported in the process, Cisco Express Forwarding, and distributed Cisco Express Forwarding forwarding paths.

Policies can be based on IPv6 address, port numbers, protocols, or packet size. For a simple policy, you can use any one of these descriptors; for a complex policy, you can use all of them.

PBR allows you to perform the following tasks:

- Classify traffic based on extended access list criteria. Access lists, then, establish the match criteria.
- Set IPv6 precedence bits, giving the network the ability to enable differentiated classes of service.
- Route packets to specific traffic-engineered paths; you might need to route them to allow a specific quality of service (QoS) through the network.

Policies can be based on IPv6 address, port numbers, protocols, or size of packets. For a simple policy, you can use any one of these descriptors; for a complex policy, you can use all of them.

PBR allows you to classify and mark packets at the edge of the network. PBR marks a packet by setting its precedence value. The precedence value can be used directly by routers in the network core to apply the appropriate QoS to a packet, which keeps packet classification at your network edge.

How Policy-Based Routing Works

All packets received on an interface with PBR enabled are passed through enhanced packet filters called route maps. The route maps used by PBR dictate the policy, determining where to forward packets.

Route maps are composed of statements. The route map statements can be marked as permit or deny, and they are interpreted in the following ways:

- If a packet matches all match statements for a route map that is marked as permit, then the device attempts to policy route the packet using the set statements. Otherwise, the packet is forwarded normally.
- If the packet matches any match statements for a route map that is marked as deny, then the packet is not subject to PBR and is forwarded normally.
- If the statement is marked as permit and the packets do not match any route map statements, the packets are sent back through the normal forwarding channels and destination-based routing is performed.

Specify PBR on the interface that receives the packet, not on the interface from which the packet is sent.

- Packet Matching, page 530
- Packet Forwarding Using Set Statements, page 531

Packet Matching

PBR for IPv6 will match packets using the **match ipv6 address** command in the associated PBR route map. Packet match criteria are those criteria supported by IPv6 access lists, as follows:

- Input interface
- Source IPv6 address (standard or extended access list [ACL])
- Destination IPv6 address (standard or extended ACL)
- Protocol (extended ACL)
- Source port and destination port (extended ACL)

- DSCP (extended ACL)
- Flow-label (extended ACL)
- Fragment (extended ACL)

Packets may also be matched by length using the match length statement in the PBR route map.

Match statements are evaluated first by the criteria specified in the **match ipv6 address** command and then by criteria specified in the **match length** command. Therefore, if both an ACL and a length statement are used, a packet will first be subject to an ACL match. Only packets that pass the ACL match will then be subject to the length match. Finally, only packets that pass both the ACL and the length statement will be policy-routed.

Packet Forwarding Using Set Statements

PBR for IPv6 packet forwarding is controlled using a number of set statements in the PBR route map. These set statements are evaluated individually in the order shown, and PBR will attempt to forward the packet using each of the of the set statements in turn. PBR evaluates each set statement by itself, without reference to any prior or subsequent set statement.

You may set multiple forwarding statements in the PBR for IPv6 route map. The following set statements may be specified:

- IPv6 next hop. The next hop to which the packet should be sent. The next hop must be present in the Routing Information Base (RIB), it must be directly connected, and it must be a global IPv6 address. If the next hop is invalid, the set statement is ignored.
- Output interface. A packet is forwarded out of a specified interface. An entry for the packet destination address must exist in the IPv6 RIB, and the specified output interface must be in the path set. If the interface is invalid, the statement is ignored.
- Default IPv6 next hop. The next hop to which the packet should be sent. It must be a global IPv6 address. This set statement is used only when there is no explicit entry for the packet destination in the IPv6 RIB.
- Default output interface. The packet is forwarded out a specified interface. This set statement is used only when there is no explicit entry for the packet destination in the IPv6 RIB.

Note The order in which PBR evaluates the set statements is the order in which they are listed above. This order may differ from the order in which route-map set statements are listed by **show** commands.

When to Use Policy-Based Routing

PBR can be used if you want certain packets to be routed some way other than the obvious shortest path. For example, PBR can be used to provide the following functionality:

- Equal access
- Protocol-sensitive routing
- Source-sensitive routing
- Routing based on interactive versus batch traffic
- Routing based on dedicated links

Some applications or traffic can benefit from QoS-specific routing; for example, you could transfer stock records to a corporate office on a higher-bandwidth, higher-cost link for a short time while sending routine application data such as e-mail over a lower-bandwidth, lower-cost link.

How to Implement Policy-Based Routing for IPv6

Enabling PBR on an Interface

To enable PBR for IPv6, you must create a route map that specifies the packet match criteria and desired policy-route action. Then you associate the route map on the required interface. All packets arriving on the specified interface that match the match clauses will be subject to PBR.

In PBR, the **set vrf** command decouples the VRF and interface association and allows the selection of a VRF based on ACL-based classification using existing PBR or route-map configurations. It provides a single router with multiple routing tables and the ability to select routes based on ACL classification. The router classifies packets based on ACL, selects a routing table, looks up the destination address, and then routes the packet.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **route-map** *map-tag* [**permit** | **deny**] [*sequence-number*
4. Do one of the following:
   - **match length** *minimum-length maximum-length*
   - **match ipv6 address** {**prefix-list** *prefix-list-name* | *access-list-name*}
5. Do one of the following:
   - **set ipv6 precedence** *precedence-value*
   - **set ipv6 next-hop** *global-ipv6-address* [*global-ipv6-address...*]
   - set interface type number [...type number]
   -
   - **set ipv6 default next-hop** *global-ipv6-address* [*global-ipv6-address...*]
   - set default interface type number [...type number]
   - **set vrf** *vrf-name*
6. **exit**
7. **interface** *type number*
8. **ipv6 policy route-map** *route-map-name*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**route-map** *map-tag* [**permit** | **deny**] [*sequence-number* **Example:** `Router(config)# route-map rip-to-ospf permit`	Defines the conditions for redistributing routes from one routing protocol into another, or enables policy routing. • Use the **route-map** command to enter route-map configuration mode.

Step 4	Do one of the following:
	• **match length** *minimum-length maximum-length*
	• **match ipv6 address** {**prefix-list** *prefix-list-name* | *access-list-name*}
	
	Example:
	`Router(config-route-map)# match length 3 200`
	
	Example:
	
	Example:
	`Router(config-route-map)# match ipv6 address marketing`
		Specifies the match criteria.
		• You can specify any or all of the following:
		◦ Matches the Level 3 length of the packet.
		◦ Matches a specified IPv6 access list.
		◦ If you do not specify a **match** command, the route map applies to all packets.

Command or Action	Purpose

Step 5 Do one of the following:
• **set ipv6 precedence** *precedence-value*
• **set ipv6 next-hop** *global-ipv6-address* [*global-ipv6-address...*]
• set interface type number [...type number]
•
• **set ipv6 default next-hop** *global-ipv6-address* [*global-ipv6-address...*]
• set default interface type number [...type number]
• **set vrf** *vrf-name*
 	
Example:
`Router(config-route-map)# set ipv6 precedence 1`
 	
Example:
 	
Example:
`Router(config-route-map)# set ipv6 next-hop 2001:DB8:2003:1::95`
 	
Example:
 	
Example:
`Router(config-route-map)# set interface GigabitEthernet 0/0/1`
 	
Example:
 	
Example:
`Router(config-route-map)# set ipv6 default next-hop 2001:DB8:2003:1::95`
	Specifies the action or actions to take on the packets that match the criteria.
	• You can specify any or all of the following:
	◦ Sets precedence value in the IPv6 header.
	◦ Sets next hop to which to route the packet (the next hop must be adjacent).
	◦ Sets output interface for the packet.
	◦ Sets next hop to which to route the packet, if there is no explicit route for this destination.
	◦ Sets output interface for the packet, if there is no explicit route for this destination.
	◦ Sets VRF instance selection within a route map for a policy-based routing VRF selection.

Command or Action	Purpose
Example:	
Example: `Router(config-route-map)# set default interface GigabitEthernet 0/0/0`	
Example:	
Example: `Router(config-route-map)# set vrf vrfname`	
Step 6 **exit** **Example:** `Router(config-route-map)# exit`	Returns the router to global configuration mode.
Step 7 **interface** *type number* **Example:** `Router(config)# interface FastEthernet 1/0`	Specifies an interface type and number, and places the router in interface configuration mode.
Step 8 **ipv6 policy route-map** *route-map-name* **Example:** `Router(config-if)# ipv6 policy-route-map interactive`	Identifies a route map to use for IPv6 PBR on an interface.

Enabling Local PBR for IPv6

Packets that are generated by the device are not normally policy routed. Perform this task to enable local PBR for IPv6 for such packets, indicating which route map the device should use.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ipv6 local policy route-map** *route-map-name*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**ipv6 local policy route-map** *route-map-name* **Example:** `Device(config)# ipv6 local policy route-map pbr-src-90`	Configures PBR for IPv6 for packets generated by the device.

Enabling Cisco Express Forwarding-Switched PBR for IPv6

No special configuration is required to enable Cisco Express Forwarding-switched PBR for IPv6. It is on by default as soon as you enable Cisco Express Forwarding and PBR on the router.

Verifying Configuration and Operation of PBR for IPv6

SUMMARY STEPS

1. **enable**
2. **show ipv6 policy**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**show ipv6 policy** **Example:** `Device# show ipv6 policy`	Displays IPv6 policy routing packet activity.

Troubleshooting PBR for IPv6

Policy routing looks at various parts of the packet and then routes the packet based on certain user-defined attributes in the packet. Perform this task to help you determine what policy routing is following, whether a packet matches the criteria, and if so, the resulting routing information for the packet.

SUMMARY STEPS

1. **enable**
2. **debug ipv6 policy** [*access-list-name*
3. **show route-map** [*map-name* | **dynamic** *dynamic-map-name* | **application** *application-name*]] | **all**] [**detailed**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**debug ipv6 policy** [*access-list-name* **Example:** `Router# debug ipv6 policy`	Displays IPv6 policy routing packet activity.

	Command or Action	Purpose
Step 3	**show route-map** [*map-name* | **dynamic** *dynamic-map-name* | **application** *application-name*]] | **all**] [**detailed** **Example:** `Router# show route-map`	Displays all route maps configured or only the one specified.

- Examples, page 538

Examples

Sample Output from the show ipv6 policy Command

The **show ipv6 policy** command displays PBR configuration, as shown in the following example:

```
Router# show ipv6 policy
Interface                 Routemap
GigabitEthernet0/0/0      src-1
```

Sample Output from the show route-map Command

The **show route-map** command displays specific route-map information, such as a count of policy matches:

```
Router# show route-map
route-map bill, permit, sequence 10
  Match clauses:
  Set clauses:
  Policy routing matches:0 packets, 0 bytes
```

Configuration Examples for Implementing Policy-Based Routing for IPv6

- Example Enabling PBR on an Interface, page 538
- Example: Enabling Local PBR for IPv6, page 539

Example Enabling PBR on an Interface

In the following example, a route map named pbr-dest-1 is created and configured, specifying packet match criteria and desired policy-route action. Then, PBR is enabled on Gigabit Ethernet interface 0/0/0.

```
ipv6 access-list match-dest-1
  permit ipv6 any 2001:DB8:2001:1760::/32
route-map pbr-dest-1 permit 10
  match ipv6 address match-dest-1
  set interface GigabitEthernet 0/0/1
interface GigabitEthernet0/0/0
  ipv6 policy-route-map interactive
```

Example: Enabling Local PBR for IPv6

In the following example, packets with a destination IPv6 address matching that allowed by access list pbr-src-90 are sent to the router at IPv6 address 2001:DB8:2003:1::95:

```
ipv6 access-list src-90
 permit ipv6 host 2001:DB8:2003::90 2001:DB8:2001:1000::/64
route-map pbr-src-90 permit 10
 match ipv6 address src-90
 set ipv6 next-hop 2001:DB8:2003:1::95
ipv6 local policy route-map pbr-src-90
```

Additional References

Related Documents

Related Topic	Document Title
IPv6 addressing and basic configuration	" Implementing IPv6 Addressing and Basic Connectivity ," *Cisco IOS IPv6 Configuration Guide*
QoS for IPv6	" Implementing QoS for IPv6 ," *Cisco IOS IPv6 Configuration Guide*
Multicast Border Gateway Protocol (BGP) for IPv6	" Implementing Multiprotocol BGP for IPv6," *Cisco IOS IPv6 Configuration Guide*
Access control lists for IPv6	" Implementing Traffic Filters and Firewalls for IPv6 Security ," *Cisco IOS IPv6 Configuration Guide*
IPv6 supported feature list	" Start Here: Cisco IOS Software Release Specifics for IPv6 Features ," *Cisco IOS IPv6 Configuration Guide*
IPv6 commands: complete command syntax, command mode, command history, defaults, usage guidelines, and examples	*Cisco IOS IPv6 Command Reference*
IPv4 quality of service	"Quality of Service Overview ," *Cisco IOS Quality of Service Solutions Configuration Guide*

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
None	To locate and download MIBs for selected platforms, Cisco software releases, and feature sets, use Cisco MIB Locator found at the following URL: http://www.cisco.com/go/mibs

RFCs

RFCs	Title
No new or modified RFCs are supported by this feature.	--

Technical Assistance

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

Feature Information for Implementing Policy-Based Routing for IPv6

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Table 24 Feature Information for Policy-Based Routing for IPv6

Feature Name	Releases	Feature Information
IPv6 Routing--IPv6 Policy-Based Routing	Cisco IOS XE Release 3.2S	Policy-based routing for IPv6 in Cisco IOS software allows a user to manually configure how received packets should be routed. The following commands were introduced or modified by this feature: **debug ipv6 policy**, **ipv6 local policy route-map**, **ipv6 policy route-map**, **match ipv6 address**, **match length**, **route-map**, **set default interface**, **set interface**, **set ipv6 default next-hop**, **set ipv6 next-hop**, **set ipv6 precedence**, **set vrf**, **show ipv6 policy**, **show route-map**

Implementing QoS for IPv6

Finding Feature Information

Your software release may not support all the features documented in this module. For the latest caveats and feature information, see Bug Search Tool and the release notes for your platform and software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the feature information table at the end of this module.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Restrictions for Implementing QoS for IPv6

The following QoS features are not supported for managing IPv6 traffic:

- Compressed Real-Time Protocol (CRTP)
- Network-based application recognition (NBAR)
- Committed access rate (CAR)
- Priority queueing (PQ)
- Custom queueing (CQ)

Information About Implementing QoS for IPv6

Implementation Strategy for QoS for IPv6

IPv6 packets are forwarded by paths that are different from those for IPv4. QoS features supported for IPv6 environments include packet classification, queueing, traffic shaping, weighted random early detection (WRED), class-based packet marking, and policing of IPv6 packets. These features are available at both the process switching and Cisco Express Forwarding switching paths of IPv6.

All of the QoS features available for IPv6 environments are managed from the modular QoS command-line interface (MQC). The MQC allows you to define traffic classes, create and configure traffic policies (policy maps), and then attach those traffic policies to interfaces.

To implement QoS in networks running IPv6, follow the same steps that you would follow to implement QoS in networks running only IPv4. At a very high level, the basic steps for implementing QoS are as follows:

- Know which applications in your network need QoS.
- Understand the characteristics of the applications so that you can make decisions about which QoS features would be appropriate.
- Know your network topology so that you know how link layer header sizes are affected by changes and forwarding.
- Create classes based on the criteria you establish for your network. In particular, if the same network is also carrying IPv4 traffic along with IPv6, decide if you want to treat both of them the same way or treat them separately and specify match criteria accordingly. If you want to treat them the same, use match statements such as **match precedence**, **match dscp**, **set precedence**, and **set dscp**. If you want to treat them separately, add match criteria such as **match protocol ip** and **match protocol ipv6** in a match-all class map.
- Create a policy to mark each class.
- Work from the edge toward the core in applying QoS features.
- Build the policy to treat the traffic.
- Apply the policy.

Packet Classification in IPv6

Packet classification is available with both the process and Cisco Express Forwarding switching path. Classification can be based on IPv6 precedence, differentiated services control point (DSCP), and other IPv6 protocol-specific values that can be specified in IPv6 access lists in addition to other non-IPv6 values such as COS, packet length, and QoS group. Once you determine which applications need QoS, you can create classes based on the characteristics of the applications. You can use a variety of match criteria to classify traffic. You can combine various match criteria to segregate, isolate, and differentiate traffic.

The enhancements to the modular QoS CLI (MQC) allow you to create matches on precedence, DSCP, and IPv6 access group values in both IPv4 and IPv6 packets. The **match** command allows matches to be made on DSCP values and precedence for both IPv4 and IPv6 packets.

Policies and Class-Based Packet Marking in IPv6 Networks

You can create a policy to mark each class of traffic with appropriate priority values, using either DSCP or precedence. Class-based marking allows you to set the IPv6 precedence and DSCP values for traffic management. The traffic is marked as it enters the router on the ingress interface. The markings are used to

treat the traffic (forward, queue) as it leaves the router on the egress interface. Always mark and treat the traffic as close as possible to its source.

Congestion Management in IPv6 Networks

Once you have marked the traffic, you can use the markings to build a policy and classify traffic on the rest of the network segments. If you keep the policy simple (e.g.,s approximately four classes), it will be easier to manage. Class-based and flow-based queueing are supported for IPv6. The processes and tasks use the same commands and arguments to configure various queueing options for both IP and IPv6.

Congestion Avoidance for IPv6 Traffic

WRED implements the RED-based drop policy on the packets that are likely to overflow the limits of class-based weighted fair queueing (CBWFQ). WRED supports class-based and flow-based (using DSCP or precedence values) queueing.

Traffic Policing in IPv6 Environments

Congestion management for IPv6 is similar to IPv4, and the commands used to configure queueing and traffic shaping features for IPv6 environments are the same commands as those used for IPv4. Traffic shaping allows you to limit the packet dequeue rate by holding additional packets in the queues and forwarding them as specified by parameters configured for traffic shaping features. Traffic shaping uses flow-based queueing by default. CBWFQ can be used to classify and prioritize the packets. Class-based policer and generic traffic shaping (GTS) or Frame Relay traffic shaping (FRTS) can be used for conditioning and policing traffic.

How to Implement QoS for IPv6

Classifying Traffic in IPv6 Networks

The **set cos** and **match cos** commands for 802.1Q (dot1Q) interfaces are supported only for Cisco Express Forwarding-switched packets. Process-switched packets, such as router-generated packets, are not marked when these options are used.

Specifying Marking Criteria for IPv6 Packets

Perform this task to establish the match criteria (or mark the packets) to be used to match packets for classifying network traffic.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **policy map** *policy-map-name*
4. **class** {*class-name* | **class-default**}
5. Do one of the following:
   - **set precedence** {*precedence-value* | *from-field* [**table** *table-map-name*]}
   - **set** [**ip**] **dscp**{*dscp-value* | *from-field* [**table** *table-map-name*]}

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables such as privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**policy map** *policy-map-name* **Example:** `Router(config)# policy map policy1`	Creates a policy map using the specified name and enters QoS policy-map configuration mode. • Enter name of policy map you want to create.
Step 4	**class** {*class-name*	**class-default**} **Example:** `Router(config-pmap)# class class-default`	Specifies the treatment for traffic of specified class (or the default class) and enters QoS policy-map class configuration mode.

	Command or Action	Purpose

Step 5	Do one of the following:
	• **set precedence** {*precedence-value* | *from-field* [**table** *table-map-name*]}
	• **set** [**ip**] **dscp**{*dscp-value* | *from-field* [**table** *table-map-name*]}
	Example:
	`Router(config-pmap-c)#`
	`set dscp cos table table-map1`
	Example:
	`Router(config-pmap-c)#`
	`set precedence cos table table-map1`
		Sets the precedence value.
		• This example is based on the CoS value (and action) defined in the specified table map.
		• Both precedence and DSCP cannot be changed in the same packets.
		• Sets the DSCP value based on the CoS value (and action) defined in the specified table map.

Using the Match Criteria to Manage IPv6 Traffic Flows

You can use multiple match statements. Depending on the type of class, you can specify whether to match all classes or any of the classes.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **class-map** {*class-name*| **class-default**}
4. Do one of the following:
   - **match precedence** *precedence-value* [*precedence-value precedence-value*]
   - **match access-group name** *ipv6-access-group*
   - **match [ip] dscp** *dscp-value* [*dscp-value dscp-value dscp-value dscp-value dscp-value dscp-value dscp-value*

DETAILED STEPS

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

	Command or Action	Purpose
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**class-map** {*class-name*| **class-default**} **Example:** `Router(config-pmap-c)# class clsl`	Creates the specified class and enters QoS class-map configuration mode.

Step 4	Do one of the following:
	• **match precedence** *precedence-value* [*precedence-value precedence-value*]
	• **match access-group name** *ipv6-access-group*
	• **match [ip] dscp** *dscp-value* [*dscp-value dscp-value dscp-value dscp-value dscp-value dscp-value dscp-value*
	
	Example:
	`Router(config-pmap-c)#`
	`match precedence 5`
	
	Example:
	`Router(config-pmap-c)# match ip dscp 15`
		Matches the precedence value. The precedence applies to both IPv4 and IPv6 packets.
		or
		Specifies the name of an IPv6 access list against whose contents packets are checked to determine if they belong to the traffic class.
		or
		Identifies a specific IP DSCP value as a match criterion.

Confirming the Service Policy

Ensure that the traffic flow matches the input or output parameter of the policy. For example, downloading a file from an FTP server generates congestion in the receive direction because the server sends large MTU-sized frames, and the client PC returns small acknowledgments (ACKs).

Before you begin this task, simulate congestion with an extended ping using a large ping size and a large number of pings. Also, try downloading a large file from an FTP server. The file constitutes "disturbing" data and fills the interface bandwidth.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type number* **multipoint** | **point-to-point**
4. **ip address ip-address mask** [*secondary*]
5. **pvc** [*name*] *vpi* / *vci* [**ces** | **ilmi** | **qsaal** | **smds**]
6. **tx-ring-limit** *ring-limit*
7. **service-policy** {**input** | **output**} *policy-map-name*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**interface** *type number* **multipoint**	**point-to-point** **Example:** `Router(config)# interface gigabitethernet1/1/0 point-to-point`	Enters interface configuration mode.
Step 4	**ip address ip-address mask** [*secondary*] **Example:** `Router(config-if)# ip address 10.1.1.1 255.255.255.0`	Specifies the IP address of the interface you want to test.

Step 5	**pvc** [*name*] *vpi* / *vci* [**ces**	**ilmi**	**qsaal**	**smds**]
				
				Example:
				`Router(config-if)#`
				`pvc cisco 0/5`
					Creates or assigns a name to an ATM PVC, optionally specifies the encapsulation type on an ATM PVC, and enters interface-ATM-VC configuration mode.

	Command or Action	Purpose
Step 6	**tx-ring-limit** *ring-limit* **Example:** `Router(config-if-atm-vc)# tx-ring-limit 10`	Reduces the size of the transmit ring of the interface. Lowering this value accelerates the use of the QoS in the Cisco IOS software. • Specify the ring limit as the number of packets for 2600 and 3600 series routers, or as the number of memory particles for 7200 and 7500 series routers.
Step 7	**service-policy** {**input** | **output**} *policy-map-name* **Example:** `Router(config-if-atm-vc)# service-policy output policy9`	Attaches a policy map to an input interface or VC, or an output interface or VC, to be used as the service policy for that interface or VC. • The packets-matched counter is a part of queueing feature and is available only on service policies attached in output direction.

Configuration Examples for Implementing QoS for IPv6

Example Verifying Cisco Express Forwarding Switching

The following is sample output from the **show cef interface detail** command for GigabitEthernet interface 1/0/0. Use this command to verify that CEF switching is enabled for policy decisions to occur. Notice that the display shows that CEF switching is enabled.

```
Router# show cef interface GigabitEthernet 1/0/0 detail

GigabitEthernet1/0/0 is up (if_number 9)
  Corresponding hwidb fast_if_number 9
  Corresponding hwidb firstsw->if_number 9
  Internet address is 10.2.61.8/24
  ICMP redirects are always sent
  Per packet load-sharing is disabled
  IP unicast RPF check is disabled
  Inbound access list is not set
  Outbound access list is not set
  IP policy routing is disabled
  Hardware idb is GigabitEthernet1/0/0
  Fast switching type 1, interface type 5
  IP Distributed CEF switching enabled
  IP Feature Fast switching turbo vector
  IP Feature CEF switching turbo vector
  Input fast flags 0x0, Output fast flags 0x0
  ifindex 7(7)
  Slot 1 Slot unit 0 VC -1
  Transmit limit accumulator 0x48001A82 (0x48001A82)
  IP MTU 1500
```

Example: Verifying Packet Marking Criteria

The following example shows how to use the **match precedence** command to manage IPv6 traffic flows:

```
Router# configure terminal
Enter configuration commands, one per line.  End with CNTL/Z.
 Router(config)# class-m c1
  Router(config-cmap)# match precedence 5
  Router(config-cmap)# end
Router#
 Router(config)# policy p1
  Router(config-pmap)# class c1
  Router(config-pmap-c)# police 10000 conform set-prec-trans 4
```

To verify that packet marking is working as expected, use the **show policy** command. The output of this command shows a difference in the number of total packets versus the number of packets marked.

```
Router# show policy p1
  Policy Map p1
    Class c1
      police 10000 1500 1500 conform-action set-prec-transmit 4 exceed-action drop
Router# configure terminal
Enter configuration commands, one per line.  End with CNTL/Z.
Router(config)# interface serial 4/1
Router(config-if)# service out p1
Router(config-if)# end
Router# show policy interface s4/1
 Serial4/1
  Service-policy output: p1
    Class-map: c1 (match-all)
      0 packets, 0 bytes
      5 minute offered rate 0 bps, drop rate 0 bps
      Match: precedence 5
      police:
        10000 bps, 1500 limit, 1500 extended limit
        conformed 0 packets, 0 bytes; action: set-prec-transmit 4
        exceeded 0 packets, 0 bytes; action: drop
        conformed 0 bps, exceed 0 bps violate 0 bps
    Class-map: class-default (match-any)
      10 packets, 1486 bytes
      5 minute offered rate 0 bps, drop rate 0 bps
      Match: any
```

During periods of transmit congestion at the outgoing interface, packets arrive faster than the interface can send them. It is helpful to know how to interpret the output of the **show policy-map interface** command, which is useful for monitoring the results of a service policy created with Cisco's MQC.

Congestion typically occurs when a fast ingress interface feeds a relatively slow egress interface. Functionally, congestion is defined as filling the transmit ring on the interface (a ring is a special buffer control structure). Every interface supports a pair of rings: a receive ring for receiving packets and a transmit ring for sending packets. The size of the rings varies with the interface controller and with the bandwidth of the interface or virtual circuit (VC). As in the following example, use the **show atm vc** *vcd* command to display the value of the transmit ring on a PA-A3 ATM port adapter.

```
Router# show atm vc 3

ATM5/0.2: VCD: 3, VPI: 2, VCI: 2
VBR-NRT, PeakRate: 30000, Average Rate: 20000, Burst Cells: 94
AAL5-LLC/SNAP, etype:0x0, Flags: 0x20, VCmode: 0x0
OAM frequency: 0 second(s)
PA TxRingLimit: 10
InARP frequency: 15 minutes(s)
Transmit priority 2
InPkts: 0, OutPkts: 0, InBytes: 0, OutBytes: 0
InPRoc: 0, OutPRoc: 0
InFast: 0, OutFast: 0, InAS: 0, OutAS: 0
```

```
InPktDrops: 0, OutPktDrops: 0
CrcErrors: 0, SarTimeOuts: 0, OverSizedSDUs: 0
OAM cells received: 0
OAM cells sent: 0
Status: UP
```

Cisco software (also referred to as the Layer 3 processor) and the interface driver use the transmit ring when moving packets to the physical media. The two processors collaborate in the following way:

- The interface sends packets according to the interface rate or a shaped rate.
- The interface maintains a hardware queue or transmit ring, where it stores the packets waiting for transmission onto the physical wire.
- When the hardware queue or transmit ring fills, the interface provides explicit back pressure to the Layer 3 processor system. It notifies the Layer 3 processor to stop dequeuing packets to the interface's transmit ring because the transmit ring is full. The Layer 3 processor now stores the excess packets in the Layer 3 queues.
- When the interface sends the packets on the transmit ring and empties the ring, it once again has sufficient buffers available to store the packets. It releases the back pressure, and the Layer 3 processor dequeues new packets to the interface.

The most important aspect of this communication system is that the interface recognizes that its transmit ring is full and throttles the receipt of new packets from the Layer 3 processor system. Thus, when the interface is congested, the drop decision is moved from a random, last-in, first-dropped decision in the first in, first out (FIFO) queue of the transmit ring to a differentiated decision based on IP-level service policies implemented by the Layer 3 processor.

Service policies apply only to packets stored in the Layer 3 queues. The table below illustrates which packets sit in the Layer 3 queue. Locally generated packets are always process switched and are delivered first to the Layer 3 queue before being passed on to the interface driver. Fast-switched and Cisco Express Forwarding-switched packets are delivered directly to the transmit ring and sit in the L3 queue only when the transmit ring is full.

Table 25 Packet Types and the Layer 3 Queue

Packet Type	Congestion	Noncongestion
Locally generated packets, including Telnet packets and pings	Yes	Yes
Other packets that are process switched	Yes	Yes
Packets that are Cisco Express Forwarding- or fast-switched	Yes	No

The following example shows these guidelines applied to the **show policy-map interface** command output.

```
Router# show policy-map interface atm 1/0.1

ATM1/0.1: VC 0/100 -
 Service-policy output: cbwfq (1283)
   Class-map: A (match-all) (1285/2)
     28621 packets, 7098008 bytes

     5 minute offered rate 10000 bps, drop rate 0 bps
     Match: access-group 101 (1289)
     Weighted Fair Queueing
       Output Queue: Conversation 73
```

```
    Bandwidth 500 (kbps) Max Threshold 64 (packets)
    (pkts matched/bytes matched) 28621/7098008

    (depth/total drops/no-buffer drops) 0/0/0
Class-map: B (match-all) (1301/4)

  2058 packets, 148176 bytes
  5 minute offered rate 0 bps, drop rate 0 bps
  Match: access-group 103 (1305)
  Weighted Fair Queueing
    Output Queue: Conversation 75
    Bandwidth 50 (kbps) Max Threshold 64 (packets)
    (pkts matched/bytes matched) 0/0
    (depth/total drops/no-buffer drops) 0/0/0
Class-map: class-default (match-any) (1309/0)
  19 packets, 968 bytes
  5 minute offered rate 0 bps, drop rate 0 bps
  Match: any  (1313)
```

The table below defines counters that appear in the example.

Table 26 Packet Counters from show policy-map interface Output

Counter	Explanation
28621 packets, 7098008 bytes	The number of packets matching the criteria of the class. This counter increments whether or not the interface is congested.
(pkts matched/bytes matched) 28621/709800	The number of packets matching the criteria of the class when the interface was congested. In other words, the interface's transmit ring was full, and the driver and the L3 processor system worked together to queue the excess packets in the L3 queues, where the service policy applies. Packets that are process switched always go through the L3 queuing system and therefore increment the "packets matched" counter.
Class-map: B (match-all) (1301/4)	These numbers define an internal ID used with the CISCO-CLASS-BASED-QOS-MIB Management Information Base (MIB).
5 minute offered rate 0 bps, drop rate 0 bps	Use the **load-interval** command to change this value and make it a more instantaneous value. The lowest value is 30 seconds; however, statistics displayed in the **show policy-map interface** command output are updated every 10 seconds. Because the command effectively provides a snapshot at a specific moment, the statistics may not reflect a temporary change in queue size.

Without congestion, there is no need to queue any excess packets. When congestion occurs, packets, including Cisco Express Forwarding- and fast-switched packets, might go into the Layer 3 queue. If you use congestion management features, packets accumulating at an interface are queued until the interface is free to send them; they are then scheduled according to their assigned priority and the queueing mechanism configured for the interface.

Normally, the packets counter is much larger than the packets matched counter. If the values of the two counters are nearly equal, then the interface is receiving a large number of process-switched packets or is heavily congested. Both of these conditions should be investigated to ensure optimal packet forwarding.

Routers allocate conversation numbers for the queues that are created when the service policy is applied. The following example shows the queues and related information.

```
Router# show policy-map interface s1/0.1 dlci 100

      Serial1/0.1: DLCI 100 -
      output : mypolicy
       Class voice
        Weighted Fair Queueing
            Strict Priority
            Output Queue: Conversation 72

              Bandwidth 16 (kbps) Packets Matched 0
             (pkts discards/bytes discards) 0/0
       Class immediate-data
        Weighted Fair Queueing
            Output Queue: Conversation 73

              Bandwidth 60 (%) Packets Matched 0
              (pkts discards/bytes discards/tail drops) 0/0/0
              mean queue depth: 0
              drops: class  random   tail     min-th   max-th   mark-prob
                     0      0        0        64       128      1/10
                     1      0        0        71       128      1/10
                     2      0        0        78       128      1/10
                     3      0        0        85       128      1/10
                     4      0        0        92       128      1/10
                     5      0        0        99       128      1/10
                     6      0        0        106      128      1/10
                     7      0        0        113      128      1/10
                     rsvp   0        0        120      128      1/10
       Class priority-data
        Weighted Fair Queueing
            Output Queue: Conversation 74

              Bandwidth 40 (%) Packets Matched 0 Max Threshold 64 (packets)
              (pkts discards/bytes discards/tail drops) 0/0/0
       Class class-default
        Weighted Fair Queueing
            Flow Based Fair Queueing
            Maximum Number of Hashed Queues 64  Max Threshold 20 (packets)
```

Information reported for each class includes the following:

- Class definition
- Queueing method applied
- Output Queue Conversation number
- Bandwidth used
- Number of packets discarded
- Number of bytes discarded
- Number of packets dropped

The **class-default**class is the default class to which traffic is directed, if that traffic does not satisfy the match criteria of other classes whose policy is defined in the policy map. The **fair-queue** command allows you to specify the number of dynamic queues into which IP flows are sorted and classified. Alternately, routers allocate a default number of queues derived from the bandwidth on the interface or VC. Supported values in either case are a power of two, in a range from 16 to 4096.

The table below lists the default values for interfaces and for ATM permanent virtual circuits (PVCs).

Table 27 Default Number of Dynamic Queues as a Function of Interface Bandwidth

Bandwidth Range	Number of Dynamic Queues
Less than or equal to 64 kbps	16
More than 64 kbps and less than or equal to 128 kbps	32
More than 128 kbps and less than or equal to 256 kbps	64
More than 256 kbps and less than or equal to 512 kbps	128
More than 512 kbps	256

The table below lists the default number of dynamic queues in relation to ATM PVC bandwidth.

Table 28 Default Number of Dynamic Queues as a Function of ATM PVC Bandwidth

Bandwidth Range	Number of Dynamic Queues
Less than or equal to 128 kbps	16
More than 128 kbps and less than or equal to 512 kbps	32
More than 512 kbps and less than or equal to 2000 kbps	64
More than 2000 kbps and less than or equal to 8000 kbps	128
More than 8000 kbps	256

Based on the number of reserved queues for WFQ, Cisco software assigns a conversation or queue number as shown in the table below.

Table 29 Conversation Numbers Assigned to Queues

Number	Type of Traffic
1 to 256	General flow-based traffic queues. Traffic that does not match to a user-created class will match to class-default and one of the flow-based queues.
257 to 263	Reserved for Cisco Discovery Protocol and for packets marked with an internal high-priority flag.

Number	Type of Traffic
264	Reserved queue for the priority class (classes configured with the priority command). Look for the "Strict Priority" value for the class in the **show policy-map** interface output. The priority queue uses a conversation ID equal to the number of dynamic queues, plus 8.
265 and higher	Queues for user-created classes.

Example Matching DSCP Value

The following example shows how to configure the service policy called priority50 and attach service policy priority50 to an interface. In this example, the **match dscp** command includes the optional **ip** keyword, meaning that the match is for IPv4 packets only. The class map called ipdscp15 will evaluate all packets entering interface Gigabit Ethernet 1/0/0. If the packet is an IPv4 packet and has a DSCP value of 15, the packet will be treated as priority traffic and will be allocated with bandwidth of 50 kbps.

```
Router(config)#
 class-map ipdscp15
Router(config-cmap)#
 match ip dscp 15
Router(config)#
 exit
Router(config)#
policy-map priority50
Router(config-pmap)#
 class ipdscp15
Router(config-pmap-c)#
priority 50
Router(config-pmap-c)#
exit
Router(config-pmap)#
 exit
Router(config)#
interface fa1/0/0
Router(config-if)#
service-policy input priority55
```

To match on IPv6 packets only, use the **match dscp** command without the **ip** keyword preceded by the **match protocol** command. Ensure that the class map has the **match-all** attribute (which is the default).

```
Router(config)#
 class-map ipdscp15
Router(config-cmap)#
 match protocol ipv6
Router(config-cmap)#
 match dscp 15
Router(config)#
 exit
```

To match packets on both IPv4 and IPv6 protocols, use the **match dscp** command:

```
Router(config)#
 class-map ipdscp15
Router(config-cmap)#
 match dscp 15
Router(config)#
 exit
```

Additional References

Related Documents

Related Topic	Document Title
IPv6 supported feature list	"Start Here: Cisco IOS Software Release Specifics for IPv6 Features ," *Cisco IOS IPv6 Configuration Guide*
IPv6 commands: complete command syntax, command mode, defaults, usage guidelines, and examples	*Cisco IOS IPv6 Command Reference*

Standards

Standard	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
None	To locate and download MIBs for selected platforms, Cisco software releases, and feature sets, use Cisco MIB Locator found at the following URL: http://www.cisco.com/go/mibs

RFCs

RFC	Title
RFC 2474	*Definition of the Differentiated Services Field (DS Field) in the IPv4 and IPv6 Headers*
RFC 2475	*An Architecture for Differentiated Services Framework*
RFC 2597	*Assured Forwarding PHB*
RFC 2598	*An Expedited Forwarding PHB*
RFC 2640	*Internet Protocol, Version 6 Specification*
RFC 2697	*A Single Rate Three Color Marker*

RFC	Title
RFC 2698	*A Two Rate Three Color Marker*

Technical Assistance

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

Feature Information for Implementing QoS for IPv6

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Table 30 Feature Information for Implementing QoS for IPv6

Feature Name	Releases	Feature Information
IPv6 Quality of Service (QoS)	Cisco IOS XE Release 2.1	QoS features supported for IPv6 environments include packet classification, queueing, traffic shaping, WRED, class-based packet marking, and policing of IPv6 packets.
IPv6 QoS--MQC Packet Marking/Re-marking	Cisco IOS XE Release 2.1	Class-based marking allows you to set the IPv6 precedence and DSCP values for traffic management.

Feature Name	Releases	Feature Information
IPv6 QoS--MQC Packet Classification	Cisco IOS XE Release 2.1 Cisco IOS XE Release 3.6S	The modular QoS CLI allows you to define traffic classes, create and configure traffic policies (policy maps), and then attach those traffic policies to interfaces. In Cisco IOS XE Release 3.6S, support was added for the Cisco ASR 903 Router.
IPv6 QoS--MQC Traffic Policing	Cisco IOS XE Release 2.1	Configuration or command usage for policing are the same in IPv6 environments as for IPv4 environments.
IPv6 QoS--MQC Traffic Shaping	Cisco IOS XE Release 2.1	Traffic shaping allows you to limit the packet dequeue rate by holding additional packets in the queues and forwarding them as specified by parameters configured for traffic shaping features.
IPv6 QoS--MQC WRED-Based Drop	Cisco IOS XE Release 2.1	WRED implements the RED-based drop policy on the packets that are likely to overflow the limits of CBWFQ.
IPv6 QoS--Queueing	Cisco IOS XE Release 2.1	Class-based and flow-based queueing are supported for IPv6.

Implementing RIP for IPv6

This module describes how to configure Routing Information Protocol for IPv6. RIP is a distance-vector routing protocol that uses hop count as a routing metric. RIP is an Interior Gateway Protocol (IGP) most commonly used in smaller networks.

Finding Feature Information

Your software release may not support all the features documented in this module. For the latest caveats and feature information, see Bug Search Tool and the release notes for your platform and software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the feature information table at the end of this module.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Information About Implementing RIP for IPv6

RIP for IPv6

IPv6 RIP functions the same and offers the same benefits as RIP in IPv4. RIP enhancements for IPv6, detailed in RFC 2080, include support for IPv6 addresses and prefixes, and the use of the all-RIP-devices multicast group address FF02::9 as the destination address for RIP update messages.

In the Cisco software implementation of IPv6 RIP, each IPv6 RIP process maintains a local routing table, referred to as a Routing Information Database (RIB). The IPv6 RIP RIB contains a set of best-cost IPv6 RIP routes learned from all its neighboring networking devices. If IPv6 RIP learns the same route from two different neighbors, but with different costs, it will store only the lowest cost route in the local RIB. The RIB also stores any expired routes that the RIP process is advertising to its neighbors running RIP. IPv6

RIP will try to insert every non-expired route from its local RIB into the master IPv6 RIB. If the same route has been learned from a different routing protocol with a better administrative distance than IPv6 RIP, the RIP route will not be added to the IPv6 RIB but the RIP route will still exist in the IPv6 RIP RIB.

Nonstop Forwarding for IPv6 RIP

Cisco nonstop forwarding (NSF) continues forwarding packets while routing protocols converge, therefore avoiding a route flap on switchover. When an RP failover occurs, the Forwarding Information Base (FIB) marks installed paths as stale by setting a new epoch. Subsequently, the routing protocols reconverge and populate the RIB and FIB. Once all NSF routing protocols converge, any stale routes held in the FIB are removed. A failsafe timer is required to delete stale routes, in case of routing protocol failure to repopulate the RIB and FIB.

RIP registers as an IPv6 NSF client. Doing so has the benefit of using RIP routes installed in the Cisco Express Forwarding table until RIP has converged on the standby.

How to Implement RIP for IPv6

Enabling IPv6 RIP

Before configuring the router to run IPv6 RIP, globally enable IPv6 using the **ipv6 unicast-routing** command in global configuration mode, and enable IPv6 on any interfaces on which IPv6 RIP is to be enabled.

If you want to set or change a global value, follow steps 1 and 2, and then use the optional **ipv6 router rip**command in global configuration mode.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ipv6 unicast-routing**
4. **interface** *type number*
5. **ipv6 rip** *name* **enable**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**ipv6 unicast-routing** **Example:** `Router(config)# ipv6 unicast-routing`	Enables the forwarding of IPv6 unicast datagrams.
Step 4	**interface** *type number* **Example:** `Router(config)# interface gigabitethernet 0/0/0`	Specifies the interface type and number, and enters interface configuration mode.
Step 5	**ipv6 rip** *name* **enable** **Example:** `Router(config-if)# ipv6 rip process1 enable`	Enables the specified IPv6 RIP routing process on an interface.

Customizing IPv6 RIP

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ipv6 router rip** *word*
4. **maximum-paths** *number-paths*
5. **exit**
6. **interface** *type number*
7. **ipv6 rip** *name* **default-information** {**only** | **originate**} [**metric** *metric-value*]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**ipv6 router rip** *word* **Example:** `Router(config)# ipv6 router rip process1`	Configures an IPv6 RIP routing process and enters router configuration mode for the IPv6 RIP routing process. • Use the *word*argument to identify a specific IPv6 RIP routing process.
Step 4	**maximum-paths** *number-paths* **Example:** `Router(config-router)# maximum-paths 1`	(Optional) Defines the maximum number of equal-cost routes that IPv6 RIP can support. • The *number-paths* argument is an integer from 1 to 64. The default for RIP is four paths.
Step 5	**exit** **Example:** `Router(config-if)# exit`	Exits interface configuration mode and enters global configuration mode.
Step 6	**interface** *type number* **Example:** `Router(config)# interface gigabitethernet 0/0/0`	Specifies the interface type and number, and enters interface configuration mode.

Command or Action	Purpose
Step 7 **ipv6 rip** *name* **default-information** {**only** | **originate**} [**metric** *metric-value*] **Example:** `Router(config-if)# ipv6 rip process1 default-information originate`	(Optional) Originates the IPv6 default route (::/0) into the specified RIP routing process updates sent out of the specified interface. **Note** To avoid routing loops after the IPv6 default route (::/0) is originated out of any interface, the routing process ignores all default routes received on any interface. • Specifying the **only** keyword originates the default route (::/0) but suppresses all other routes in the updates sent on this interface. • Specifying the **originate** keyword originates the default route (::/0) in addition to all other routes in the updates sent on this interface.

Redistributing Routes into an IPv6 RIP Routing Process

The maximum metric that RIP can advertise is 16, and a metric of 16 denotes a route that is unreachable. Therefore, if you are redistributing routes with metrics greater than or equal to 16, then by default RIP will advertise them as unreachable. These routes will not be used by neighboring routers. The user must configure a redistribution metric of less than 15 for these routes.

Note You must to advertise a route with metric of 15 or less. A RIP router always adds an interface cost--the default is 1--onto the metric of a received route. If you advertise a route with metric 15, your neighbor will add 1 to it, making a metric of 16. Because a metric of 16 is unreachable, your neighbor will not install the route in the routing table.

If no metric is specified, then the current metric of the route is used. To find the current metric of the route, enter the **show ipv6 route** command.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type number*
4. **ipv6 rip** *word* **enable**
5. **redistribute** *protocol* [*process-id*] {**level-1** | **level-1-2**| **level-2**} [**metric** *metric-value*] [**metric-type**{**internal** | **external**}] [**route-map** *map-name*]

DETAILED STEPS

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

	Command or Action	Purpose
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**interface** *type number* **Example:** `Router(config)# interface gigabitethernet 0/0/0`	Specifies the interface type and number, and enters interface configuration mode.
Step 4	**ipv6 rip** *word* **enable** **Example:** `Router(config-if)# ipv6 router one enable`	Enables an IPv6 Routing Information Protocol (RIP) routing process on an interface.
Step 5	**redistribute** *protocol* [*process-id*] {**level-1** | **level-1-2**| **level-2**} [**metric** *metric-value*] [**metric-type**{**internal** | **external**}] [**route-map** *map-name*] **Example:** `Router(config-router)# redistribute bgp 65001 route-map bgp-to-rip`	Redistributes the specified routes into the IPv6 RIP routing process. • The *protocol* argument can be one of the following keywords: **bgp**, **connected**, **isis**, **rip**, or **static**. • The **rip** keyword and *process-id* argument specify an IPv6 RIP routing process. **Note** The **connected** keyword refers to routes that are established automatically by assigning IPv6 addresses to an interface.

Configuring Route Tags for IPv6 RIP Routes

When performing route redistribution, you can associate a numeric tag with a route. The tag is advertised with the route by RIP and will be installed along with the route in neighboring router's routing table.

If you redistribute a tagged route (for example, a route in the IPv6 routing table that already has a tag) into RIP, then RIP will automatically advertise the tag with the route. If you use a redistribution route map to specify a tag, then RIP will use the route map tag in preference to the routing table tag.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **route-map** *map-tag* [**permit** | **deny**] [*sequence-number*]
4. **match ipv6 address** {**prefix-list** *prefix-list-name* | *access-list-name*
5. **set tag** *tag-value*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**route-map** *map-tag* [**permit** | **deny**] [*sequence-number*] **Example:** `Router(config)# route-map bgp-to-rip permit 10`	Defines a route map, and enters route-map configuration mode. • Follow this step with a **match** command.
Step 4	**match ipv6 address** {**prefix-list** *prefix-list-name* | *access-list-name* **Example:** `Router(config-route-map)# match ipv6 address prefix-list bgp-to-rip-flt`	Specifies a list of IPv6 prefixes to be matched.
Step 5	**set tag** *tag-value* **Example:** `Router(config-route-map)# set tag 4`	Sets the tag value to associate with the redistributed routes.

Filtering IPv6 RIP Routing Updates

Route filtering using distribute lists provides control over the routes RIP receives and advertises. This control may be exercised globally or per interface.

Filtering is controlled by distribute lists. Input distribute lists control route reception, and input filtering is applied to advertisements received from neighbors. Only those routes that pass input filtering will be inserted in the RIP local routing table and become candidates for insertion into the IPv6 routing table.

Output distribute lists control route advertisement; Output filtering is applied to route advertisements sent to neighbors. Only those routes passing output filtering will be advertised.

Global distribute lists (which are distribute lists that do not apply to a specified interface) apply to all interfaces. If a distribute list specifies an interface, then that distribute list applies only to that interface.

An interface distribute list always takes precedence. For example, for a route received at an interface, with the interface filter set to deny, and the global filter set to permit, the route is blocked, the interface filter is passed, the global filter is blocked, and the route is passed.

IPv6 prefix lists are used to specify certain prefixes or a range of prefixes that must be matched before a permit or deny statement can be applied. Two operand keywords can be used to designate a range of prefix lengths to be matched. A prefix length of less than, or equal to, a value is configured with the **le**keyword. A prefix length greater than, or equal to, a value is specified using the **ge** keyword. The **ge** and **le** keywords can be used to specify the range of the prefix length to be matched in more detail than the usual *ipv6-prefix* / *prefix-length* argument. For a candidate prefix to match against a prefix list entry three conditions can exist:

- The candidate prefix must match the specified prefix list and prefix length entry.
- The value of the optional **le** keyword specifies the range of allowed prefix lengths from the *prefix-length* argument up to, and including, the value of the **le** keyword.
- The value of the optional **ge** keyword specifies the range of allowed prefix lengths from the value of the **ge** keyword up to, and including, 128.

Note Note that the first condition must match before the other conditions take effect.

An exact match is assumed when the **ge** or **le** keywords are not specified. If only one keyword operand is specified then the condition for that keyword is applied, and the other condition is not applied. The *prefix-length* value must be less than the **ge** value. The **ge** value must be less than, or equal to, the **le** value. The **le** value must be less than or equal to 128.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ipv6 prefix list** *prefix-list-name* **seq** *seq-number*] {**deny** *ipv6-prefix*/*prefix-length* | **description** *text*} [**ge** *ge-value*] [**le** *le-value*
4. **ipv6 prefix list** *prefix-list-name* **seq** *seq-number*] {**deny** *ipv6-prefix*/*prefix-length* | **description** *text*} [**ge** *ge-value*] [**le** *le-value*
5. Repeat Steps 3 and 4 as many times as necessary to build the prefix list.
6. **ipv6 router rip** *name*
7. **distribute-list prefix-list** *prefix-list-name* **in** | **out**} [*interface-type interface-number*

DETAILED STEPS

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

	Command or Action	Purpose
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**ipv6 prefix list** *prefix-list-name* **seq** *seq-number*] {**deny** *ipv6-prefix*/*prefix-length* | **description** *text*} [**ge** *ge-value*] [**le** *le-value* **Example:** `Router(config)# ipv6 prefix-list abc permit 2001:DB8::/16`	Creates an entry in the IPv6 prefix list.
Step 4	**ipv6 prefix list** *prefix-list-name* **seq** *seq-number*] {**deny** *ipv6-prefix*/*prefix-length* | **description** *text*} [**ge** *ge-value*] [**le** *le-value* **Example:** `Router(config)# ipv6 prefix-list abc deny ::/0`	Creates an entry in the IPv6 prefix list.
Step 5	Repeat Steps 3 and 4 as many times as necessary to build the prefix list.	--
Step 6	**ipv6 router rip** *name* **Example:** `Router(config)# ipv6 router rip process1`	Configures an IPv6 RIP routing process.
Step 7	**distribute-list prefix-list** *prefix-list-name* **in** | **out**} [*interface-type interface-number* **Example:** `Router(config-rtr-rip)# distribute-list prefix-list process1 in gigabitethernet 0/0/0`	Applies a prefix list to IPv6 RIP routing updates that are received or sent on an interface.

Verifying IPv6 RIP Configuration and Operation

SUMMARY STEPS

1. **show ipv6 rip** [*name*][**database**| **next-hops**]
2. **show ipv6 route** [*ipv6-address*| *ipv6-prefix*/*prefix-length*| *protocol* | *interface-type interface-number*]
3. **enable**
4. **debug ipv6 rip** [*interface-type interface-number*]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**show ipv6 rip** [*name*][**database**| **next-hops**] **Example:** `Router> show ipv6 rip process1 database`	(Optional) Displays information about current IPv6 RIP processes. • In this example, IPv6 RIP process database information is displayed for the specified IPv6 RIP process.
Step 2	**show ipv6 route** [*ipv6-address*| *ipv6-prefix*/*prefix-length*| *protocol* | *interface-type interface-number*] **Example:** `Router> show ipv6 route rip`	(Optional) Displays the current contents of the IPv6 routing table. • In this example, only IPv6 RIP routes are displayed.
Step 3	**enable** **Example:** `Router> enable`	Enables higher privilege levels, such as privileged EXEC mode. • Enter your password if prompted.
Step 4	**debug ipv6 rip** [*interface-type interface-number*] **Example:** `Router# debug ipv6 rip`	(Optional) Displays debugging messages for IPv6 RIP routing transactions.

- Examples, page 570

Examples

- Sample Output for the show ipv6 rip Command, page 570
- Sample Output for the show ipv6 route Command, page 571
- Sample Output for the debug ipv6 rip Command, page 571

Sample Output for the show ipv6 rip Command

In the following example, output information about all current IPv6 RIP processes is displayed using the **show ipv6 rip**command:

```
Router> show ipv6 rip
RIP process "process1", port 521, multicast-group FF02::9, pid 62
     Administrative distance is 120. Maximum paths is 1
     Updates every 5 seconds, expire after 15
     Holddown lasts 10 seconds, garbage collect after 30
     Split horizon is on; poison reverse is off
     Default routes are generated
     Periodic updates 223, trigger updates 1
  Interfaces:
```

```
    Gigabitethernet0/0/0
  Redistribution:
    Redistributing protocol bgp 65001 route-map bgp-to-rip
```

In the following example, output information about a specified IPv6 RIP process database is displayed using the **show ipv6 rip**command with the *name* argument and the **database** keyword. In the following output for the IPv6 RIP process named process1, timer information is displayed, and route 2001:DB8::16/64 has a route tag set:

```
Router> show ipv6 rip process1 database
RIP process "process1", local RIB
 2001:DB8::/64, metric 2
     Gigabitethernet0/0/0/FE80::A8BB:CCFF:FE00:B00, expires in 13 secs
 2001:DB8::/16, metric 2 tag 4, installed
     Gigabitethernet0/0/0/FE80::A8BB:CCFF:FE00:B00, expires in 13 secs
 2001:DB8:1::/16, metric 2 tag 4, installed
     Gigabitethernet0/0/0/FE80::A8BB:CCFF:FE00:B00, expires in 13 secs
 2001:DB8:2::/16, metric 2 tag 4, installed
     Gigabitethernet0/0/0/FE80::A8BB:CCFF:FE00:B00, expires in 13 secs
 ::/0, metric 2, installed
     Gigabitethernet0/0/0FE80::A8BB:CCFF:FE00:B00, expires in 13 secs
```

In the following example, output information for a specified IPv6 RIP process is displayed using the **show ipv6 rip**user EXEC command with the *name* argument and the **next-hops** keyword:

```
Router> show ipv6 rip process1 next-hops
RIP process "process1", Next Hops
  FE80::A8BB:CCFF:FE00:A00/Gigabitethernet0/0/0 [4 paths]
```

Sample Output for the show ipv6 route Command

The current metric of the route can be found by entering the **show ipv6 route** command. In the following example, output information for all IPv6 RIP routes is displayed using the **show ipv6 route**command with the **rip** protocol keyword:

```
Router> show ipv6 route rip
IPv6 Routing Table - 17 entries
Codes: C - Connected, L - Local, S - Static, R - RIP, B - BGP
       U - Per-user Static route
       I1 - ISIS L1, I2 - ISIS L2, IA - ISIS interarea
       O - OSPF intra, OI - OSPF inter, OE1 - OSPF ext 1, OE2 - OSPF ext 2
R   2001:DB8:1::/32 [120/2]
     via FE80::A8BB:CCFF:FE00:A00, gigabitethernet0/0/0
R   2001:DB8:2::/32 [120/2]
     via FE80::A8BB:CCFF:FE00:A00, gigabitethernet0/0/0
R   2001:DB8:3::/32 [120/2]
     via FE80::A8BB:CCFF:FE00:A00, gigabitethernet0/0/0
```

Sample Output for the debug ipv6 rip Command

In the following example, debugging messages for IPv6 RIP routing transactions are displayed using the **debug ipv6 rip**command:

```
Router# debug ipv6 rip
RIPng: Sending multicast update on gigabitethernet0/0/0 for process1
       src=FE80::A8BB:CCFF:FE00:B00
       dst=FF02::9 (gigabitethernet0/0/0)
       sport=521, dport=521, length=112
       command=2, version=1, mbz=0, #rte=5
       tag=0, metric=1, prefix=2001:DB8::/64
       tag=4, metric=1, prefix=2001:DB8:1::/16
       tag=4, metric=1, prefix=2001:DB8:2;:/16
       tag=4, metric=1, prefix=2001:DB8:3::/16
       tag=0, metric=1, prefix=::/0
RIPng: Next RIB walk in 10032
```

```
RIPng: response received from FE80::A8BB:CCFF:FE00:A00 on gigabitethernet0/0/0 for
process1
       src=FE80::A8BB:CCFF:FE00:A00 (gigabitethernet0/0/0)
       dst=FF02::9
       sport=521, dport=521, length=92
       command=2, version=1, mbz=0, #rte=4
       tag=0, metric=1, prefix=2001:DB8::/64
       tag=0, metric=1, prefix=2001:DB8:1::/32
       tag=0, metric=1, prefix=2001:DB8:2::/32
       tag=0, metric=1, prefix=2001:DB8:3::/32
```

Configuration Examples for IPv6 RIP

Example IPv6 RIP Configuration

In the following example, the IPv6 RIP process named process1 is enabled on the router and on Gigabit Ethernet interface 0/0/0. The IPv6 default route (::/0) is advertised in addition to all other routes in router updates sent on Gigabit Ethernet interface 0/0/0. Additionally, BGP routes are redistributed into the RIP process named process1 according to a route map where routes that match a prefix list are also tagged. The number of parallel paths is set to one to allow the route tagging, and the IPv6 RIP timers are adjusted. A prefix list named eth0/0-in-flt filters inbound routing updates on Gigabit Ethernet interface 0/0/0.

```
ipv6 router rip process1
 maximum-paths 1
 redistribute bgp 65001 route-map bgp-to-rip
 distribute-list prefix-list eth0/0-in-flt in Gigabitethernet0/0/0
!
interface Gigabitethernet0/0/0
 ipv6 address 2001:DB8::/64 eui-64
 ipv6 rip process1 enable
 ipv6 rip process1 default-information originate
!
ipv6 prefix-list bgp-to-rip-flt seq 10 deny 2001:DB8:3::/16 le 128
ipv6 prefix-list bgp-to-rip-flt seq 20 permit 2001:DB8:1::/8 le 128
!
ipv6 prefix-list eth0/0-in-flt seq 10 deny ::/0
ipv6 prefix-list eth0/0-in-flt seq 15 permit ::/0 le 128
!
route-map bgp-to-rip permit 10
 match ipv6 address prefix-list bgp-to-rip-flt
 set tag 4
```

Additional References

Related Documents

Related Topic	Document Title
IPv4 RIP configuration tasks	" Configuring Routing Information Protocol ," *Cisco IOS XE IP Routing Protocols Configuration Guide*

Related Topic	Document Title
RIP commands: complete command syntax, command mode, defaults, usage guidelines, and examples	" RIP Commands ," *Cisco IOS IP Routing Protocols Command Reference*
IPv6 supported feature list	"Start Here: Cisco IOS XE Software Release Specifics for IPv6 Features ," *Cisco IOS XE IPv6 Configuration Guide*
IPv6 commands: complete command syntax, command mode, defaults, usage guidelines, and examples	*Cisco IOS IPv6 Command Reference*
Cisco IOS master command list, all releases	Cisco IOS Master Command List, All Releases

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
None	To locate and download MIBs for selected platforms, Cisco software releases, and feature sets, use Cisco MIB Locator found at the following URL: http://www.cisco.com/go/mibs

RFCs

RFCs	Title
RFC 2080	*RIPng for IPv6*

Technical Assistance

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

Feature Information for Implementing RIP for IPv6

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Table 31 Feature Information for Implementing RIP for IPv6

Feature Name	Releases	Feature Information
IPv6--RIPng Nonstop Forwarding	Cisco IOS XE Release 2.1	IPv6 RIP supports NSF.
IPv6 Routing--RIP for IPv6 (RIPng)	Cisco IOS XE Release 2.1	RIP enhancements for IPv6 include support for IPv6 addresses and prefixes, and the use of the all-RIP-routers multicast group address FF02::9 as the destination address for RIP update messages. The following commands were modified by this feature: **debug ipv6 rip**, **ipv6 rip default-information**, **ipv6 rip enable**, **ipv6 router rip**, **ipv6 unicast-routing, maximum paths**, **distribute-list prefix-list (IPv6 RIP)**, **ipv6 prefix-list**, **show ipv6 rip**, **timers (IPv6 RIP)**

Feature Name	Releases	Feature Information
IPv6 Routing--Route Redistribution	Cisco IOS XE Release 2.1	Routes may be specified by prefix, using a route-map prefix list, or by tag, using the route-map "match tag" function. The following commands were modified by this feature: **ipv6 rip enable**, **match ipv6 address**, **redistribute,**route-map, set tag, show ipv6 route

Implementing Selective Packet Discard in IPv6

First Published: August 21, 2007

Last Updated: November 18, 2010

This document describes the Selective Packet Discard (SPD) feature in IPv6. The SPD feature in IPv6 manages the process level input queues on the Route Processor (RP). SPD provides priority to routing protocol packets and other important traffic control Layer 2 keepalives during periods of process level queue congestion.

Finding Feature Information

Your software release may not support all the features documented in this module. For the latest caveats and feature information, see Bug Search Tool and the release notes for your platform and software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the feature information table at the end of this module.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Information About Implementing Selective Packet Discard in IPv6

SPD in IPv6 Overview

The SPD mechanism manages the process level input queues on the RP. SPD provides priority to routing protocol packets and other important traffic control Layer 2 keepalives during periods of process level queue congestion.

SPD State Check

The SPD state check is performed on the IPv6 process input queue on the RP. High-priority packets, such as those of IP precedence 7, are not applied to SPD and are never dropped. All remaining packets, however, can be dropped depending on the length of the IPv6 packet input queue and the SPD state. The possible SPD states are as follows:

- Normal: The process input queue is less than the SPD minimum threshold.
- Random drop: The process input queue is between the SPD minimum and maximum thresholds.
- Max: The process input queue is equal to the SPD maximum threshold.

The size of the process input queue governs the SPD state: normal (no drop), random drop, or max. When the process input queue is less than the SPD minimum threshold, SPD takes no action and enters normal state. In the normal state, no packets are dropped. When the input queue reaches the maximum threshold, SPD enters max state, in which normal priority packets are discarded. If the input queue is between the minimum and maximum thresholds, SPD enters the random drop state, in which normal packets may be dropped.

SPD Mode

Three IPv6 SPD modes are supported: none (which is the default), aggressive drop, and OSPF mode. The aggressive drop mode discards incorrectly formatted packets when the IPv6 is in the random drop state. OSPF mode provides a mechanism whereby OSPF packets are handled with SPD priority.

SPD Headroom

With SPD, the behavior of normal IPv6 packets is not changed. However, routing protocol packets are given higher priority, because SPD recognizes routing protocol packets by the IPv6 precedence field. Therefore, if the IPv6 precedence is set to 7, then the packet is given priority.

SPD prioritizes IPv6 packets with a precedence of 7 by allowing the Cisco IOS software to queue them into the process level input queue above the normal input queue limit. The number of packets allowed in excess of the normal limit is called the SPD headroom. The SPD headroom default is 100, which means that a high precedence packet is not dropped if the size of the input hold queue is lower than 175 (which is the input queue default size + SPD headroom size).

Because Interior Gateway Protocols (IGPs) and link stability are tenuous and crucial, such packets are given the highest priority and are given extended SPD headroom with a default of 10 packets. These packets are not dropped if the size of the input hold queue is lower than 185 (input queue default size + SPD headroom size + SPD extended headroom).

Non-IPv6 packets such as Connectionless Network Service Intermediate System-to-Intermediate System (CLNS IS-IS) packets, PPP packets, and High-Level Data Link Control (HDLC) keepalives are treated as normal priority as a result of being Layer 2 instead of Layer 3. In addition, IGPs operating at Layer 3 or higher are given priority over normal IPv6 packets, but are given the same priority as Border Gateway Protocol (BGP) packets. Therefore, during BGP convergence or during times of very high BGP activity, IGP hellos and keepalives often are dropped, causing IGP adjacencies to fail.

How to Implement Selective Packet Discard in IPv6

Configuring the SPD Process Input Queue

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ipv6 spd queue max-threshold** *value*
4. **ipv6 spd queue min-threshold** *value*
5. exit
6. **show ipv6 spd**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**ipv6 spd queue max-threshold** *value* **Example:** `Router(config)# ipv6 spd queue max-threshold 60000`	Configures the maximum number of packets in the SPD process input queue.
Step 4	**ipv6 spd queue min-threshold** *value* **Example:** `Router(config)# ipv6 spd queue max-threshold 4094`	Configures the minimum number of packets in the IPv6 SPD process input queue.

	Command or Action	Purpose
Step 5	exit **Example:** `Router(config)# exit`	Returns the router to privileged EXEC mode.
Step 6	**show ipv6 spd** **Example:** `Router# show ipv6 spd`	Displays IPv6 SPD configuration.

Configuring SPD Mode

No IPv6 SPD mode is configured by default. However, you may want to configure the router to use a specific mode when the router enters a specified IPv6 SPD state or to prioritize certain packets.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ipv6 spd mode** {**aggressive** | **ospf**}

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**ipv6 spd mode** {**aggressive**	**ospf**} **Example:** `Router(config)# ipv6 spd mode aggressive`	Configures an IPv6 SPD mode.

Configuring SPD Headroom

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **spd headroom** *size*
4. **spd extended-headroom** *size*
5. exit
6. **show ipv6 spd**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**spd headroom** *size* **Example:** `Router(config)# spd headroom 200`	Configures SPD headroom.
Step 4	**spd extended-headroom** *size* **Example:** `Router(config)# spd extended-headroom 11`	Configures extended SPD headroom.
Step 5	exit **Example:** `Router(config)# exit`	Returns the router to privileged EXEC mode.

	Command or Action	Purpose
Step 6	**show ipv6 spd** **Example:** `Router# show ipv6 spd`	Displays the IPv6 SPD configuration.

Configuration Examples for IPv6 Selective Packet Discard

Example: Configuring the SPD Process Input Queue

The following example shows the SPD process input queue configuration. The maximum process input queue threshold is 60,000, and the SPD state is normal. The headroom and extended headroom values are the default:

```
Router# ipv6 spd queue max-threshold 5000
Router# show ipv6 spd

Current mode: normal
Queue max threshold: 60000, Headroom: 100, Extended Headroom: 10
IPv6 packet queue: 0
```

Additional References

Related Documents

Related Topic	Document Title
IPv6 supported feature list	" Start Here: Cisco IOS Software Release Specifics for IPv6 Features ," *Cisco IOS XE IPv6 Configuration Guide*
IPv6 commands: complete command syntax, command mode, defaults, usage guidelines, and examples	*Cisco IOS IPv6 Command Reference*
Cisco IOS master command list, all releases	Cisco IOS Master Command List, All Releases

Standards

Standard	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
None	To locate and download MIBs for selected platforms, Cisco software releases, and feature sets, use Cisco MIB Locator found at the following URL: http://www.cisco.com/go/mibs

RFCs

RFC	Title
RFC 2474	*Definition of the Differentiated Services Field (DS Field) in the IPv4 and IPv6 Headers*

Technical Assistance

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

Feature Information for Implementing Selective Packet Discard in IPv6

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Table 32 Feature Information for Implementing Selective Packet Discard in IPv6

Feature Name	Releases	Feature Information
IPv6 Selective Packet Discard	Cisco IOS XE Release 2.6	The SPD mechanism manages the process level input queues on the RP. SPD provides priority to routing protocol packets and other important traffic control Layer 2 keepalives during periods of process level queue congestion. The following commands were introduced or modified: **ipv6 spd mode**, **ipv6 spd queue max-threshold**, **ipv6 spd queue min-threshold**, **show ipv6 spd**, **spd extended-headroom**, **spd headroom**.

Implementing Static Routes for IPv6

This module describes how to configure static routes for IPv6. Routing defines the paths over which packets travel in the network. Manually configured static routes may be used instead of dynamic routing protocols for smaller networks or for sections of a network that have only one path to an outside network. Lack of redundancy limits the usefulness of static routes, and in larger networks manual reconfiguration of routes can become a large administrative overhead.

Finding Feature Information

Your software release may not support all the features documented in this module. For the latest caveats and feature information, see Bug Search Tool and the release notes for your platform and software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the feature information table at the end of this module.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Information About Implementing Static Routes for IPv6

Static Routes

Networking devices forward packets using route information that is either manually configured or dynamically learned using a routing protocol. Static routes are manually configured and define an explicit path between two networking devices. Unlike a dynamic routing protocol, static routes are not

automatically updated and must be manually reconfigured if the network topology changes. The benefits of using static routes include security and resource efficiency. Static routes use less bandwidth than dynamic routing protocols and no CPU cycles are used to calculate and communicate routes. The main disadvantage to using static routes is the lack of automatic reconfiguration if the network topology changes.

Static routes can be redistributed into dynamic routing protocols but routes generated by dynamic routing protocols cannot be redistributed into the static routing table. No algorithm exists to prevent the configuration of routing loops that use static routes.

Static routes are useful for smaller networks with only one path to an outside network and to provide security for a larger network for certain types of traffic or links to other networks that need more control. In general, most networks use dynamic routing protocols to communicate between networking devices but may have one or two static routes configured for special cases.

Directly Attached Static Routes

In directly attached static routes, only the output interface is specified. The destination is assumed to be directly attached to this interface, so the packet destination is used as the next-hop address. This example shows such a definition:

```
ipv6 route 2001:DB8::/32 gigabitethernet1/0/0
```

The example specifies that all destinations with address prefix 2001:DB8::/32 are directly reachable through interface GigabitEthernet1/0/0.

Directly attached static routes are candidates for insertion in the IPv6 routing table only if they refer to a valid IPv6 interface; that is, an interface that is both up and has IPv6 enabled on it.

Recursive Static Routes

In a recursive static route, only the next hop is specified. The output interface is derived from the next hop. This example shows such a definition:

```
ipv6 route 2001:DB8::/32 2001:DB8:3000:1
```

This example specifies that all destinations with address prefix 2001:DB8::/32 are reachable via the host with address 2001:DB8:3000:1.

A recursive static route is valid (that is, it is a candidate for insertion in the IPv6 routing table) only when the specified next hop resolves, either directly or indirectly, to a valid IPv6 output interface, provided the route does not self-recurse, and the recursion depth does not exceed the maximum IPv6 forwarding recursion depth.

A route self-recurses if it is itself used to resolve its own next hop. For example, suppose we have the following routes in the IPv6 routing table:

```
IPv6 Routing Table - 9 entries
Codes: C - Connected, L - Local, S - Static, R - RIP, B - BGP
       U - Per-user Static route
       I1 - ISIS L1, I2 - ISIS L2, IA - ISIS interarea
       O - OSPF intra, OI - OSPF inter, OE1 - OSPF ext 1, OE2 - OSPF ext 2
R   2001:DB8::/32 [130/0]
     via ::, Serial2/0
B   2001:DB8:3000:0/16 [200/45]
     Via 2001:DB8::0104
```

The following examples defines a recursive IPv6 static route:

```
ipv6 route
2001:DB8::/32 2001:0BD8:3000:1
```

This static route will not be inserted into the IPv6 routing table because it is self-recursive. The next hop of the static route, 2001:DB8:3000:1, resolves via the BGP route 2001:DB8:3000:0/16, which is itself a recursive route (that is, it only specifies a next hop). The next hop of the BGP route, 2001:DB8::0104, resolves via the static route. Therefore, the static route would be used to resolve its own next hop.

It is not normally useful to manually configure a self-recursive static route, although it is not prohibited. However, a recursive static route that has been inserted in the IPv6 routing table may become self-recursive as a result of some transient change in the network learned through a dynamic routing protocol. If this occurs, the fact that the static route has become self-recursive will be detected and it will be removed from the IPv6 routing table, although not from the configuration. A subsequent network change may cause the static route to no longer be self-recursive, in which case it will be reinserted in the IPv6 routing table.

Fully Specified Static Routes

In a fully specified static route, both the output interface and the next hop are specified. This form of static route is used when the output interface is a multi-access one and it is necessary to explicitly identify the next hop. The next hop must be directly attached to the specified output interface. The following example shows a definition of a fully specified static route:

```
ipv6 route 2001:DB8:/32 gigabitethernet1/0/0 2001:DB8:3000:1
```

A fully specified route is valid (that is, a candidate for insertion into the IPv6 routing table) when the specified IPv6 interface is IPv6-enabled and up.

Floating Static Routes

Floating static routes are static routes that are used to back up dynamic routes learned through configured routing protocols. A floating static route is configured with a higher administrative distance than the dynamic routing protocol it is backing up. As a result, the dynamic route learned through the routing protocol is always used in preference to the floating static route. If the dynamic route learned through the routing protocol is lost, the floating static route will be used in its place. The following example defines a floating static route:

```
ipv6 route 2001:DB8:/32 gigabitethernet1/0/0 2001:DB8:3000:1 210
```

Any of the three types of IPv6 static routes can be used as a floating static route. A floating static route must be configured with an administrative distance that is greater than the administrative distance of the dynamic routing protocol, because routes with smaller administrative distances are preferred.

Note By default, static routes have smaller administrative distances than dynamic routes, so static routes will be used in preference to dynamic routes.

How to Implement Static Routes for IPv6

- Configuring a Static IPv6 Route, page 588

Configuring a Static IPv6 Route

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ipv6 route** *ipv6-prefix / prefix-length ipv6-address | interface-type interface-number ipv6-address*]} [*administrative-distance*] [*administrative-multicast-distance* | **unicast**| **multicast**] [**tag** *tag*]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**ipv6 route** *ipv6-prefix / prefix-length ipv6-address	interface-type interface-number ipv6-address*]} [*administrative-distance*] [*administrative-multicast-distance*	**unicast**| **multicast**] [**tag** *tag*] **Example:** `Device(config)# ipv6 route ::/0 serial 2/0`	Configures a static IPv6 route. • A static default IPv6 route is being configured on a serial interface. • See the syntax examples that immediately follow this table for specific uses of the **ipv6 route** command for configuring static routes.

Configuring a Floating Static IPv6 Route

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ipv6 route** *ipv6-prefix / prefix-length* {*ipv6-address | interface-type interface-number ipv6-address*]} [*administrative-distance*] [*administrative-multicast-distance* | **unicast** | **multicast**] [**tag** *tag*]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**ipv6 route** *ipv6-prefix* / *prefix-length* {*ipv6-address* | *interface-type interface-number ipv6-address*]} [*administrative-distance*] [*administrative-multicast-distance* | **unicast** | **multicast**] [**tag** *tag*] **Example:** `Device(config)# ipv6 route 2001:DB8::/32 serial 2/0 201`	Configures a static IPv6 route. • In this example, a floating static IPv6 route is being configured. • Default administrative distances are as follows: ◦ Connected interface--0 ◦ Static route--1 ◦ Enhanced Interior Gateway Routing Protocol (EIGRP) summary route--5 ◦ External Border Gateway Protocol (eBGP)--20 ◦ Internal Enhanced IGRP--90 ◦ IGRP--100 ◦ Open Shortest Path First--110 ◦ Intermediate System-to-Intermediate System (IS-IS)--115 ◦ Routing Information Protocol (RIP)--120 ◦ Exterior Gateway Protocol (EGP)--140 ◦ EIGRP external route--170 ◦ Internal BGP--200 ◦ Unknown--255

Verifying Static IPv6 Route Configuration and Operation

SUMMARY STEPS

1. **enable**
2. Do one of the following:
   - **show ipv6 static** [*ipv6-address* | *ipv6-prefix*/*prefix-length*][**interface** *interface-type interface-number*] [**recursive**] [**detail**]
   -
   -
   - **show ipv6 route** [*ipv6-address* | *ipv6-prefix*/*prefix-length* | *protocol* | *interface-type interface-number*
3. **debug ipv6 routing**

DETAILED STEPS

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

Command or Action	Purpose

Step 2 Do one of the following:
• **show ipv6 static** [*ipv6-address* | *ipv6-prefix*/*prefix-length*] [**interface** *interface-type interface-number*] [**recursive**] [**detail**]
•
•
• **show ipv6 route** [*ipv6-address* | *ipv6-prefix*/*prefix-length* | *protocol* | *interface-type interface-number*
Example:
`Router# show ipv6 static`
Example:
Example:
Example:
`Router# show ipv6 route static`
	Displays the current contents of the IPv6 routing table.
	• These examples show two different ways of displaying IPv6 static routes.
	• Refer to the **show ipv6 static** and **show ipv6 route** command entries in the Cisco IOS IPv6 Command Reference for more details on the arguments and keywords used in this command.

Step 3 **debug ipv6 routing** **Example:** `Router# debug ipv6 routing`	Displays debugging messages for IPv6 routing table updates and route cache updates.

- Examples, page 591

Examples

- Sample Output from the ipv6 route Command, page 592
- Sample Output from the show ipv6 static Command When No Options Are Specified in the Command Syntax, page 592
- Sample Output from the show ipv6 static Command with the IPv6 Address and Prefix Command, page 592
- Sample Output from the show ipv6 static interface Command, page 592
- Sample Output from the show ipv6 static recursive Command, page 593
- Sample Output from the show ipv6 static detail Command, page 593
- Sample Output from the show ipv6 route Command, page 593

- Sample Output from the debug ipv6 routing Command, page 594

Sample Output from the ipv6 route Command

In addition to the syntax example included in the Sample Output from the ipv6 route Command, page 592, the following syntax examples illustrate use of the **ipv6 route** for configuring the various types of static routes.

The following example shows how to configure a directly attached static route through a point-to-point interface.

```
Router(config)# ipv6 route 2001:DB8::/32 serial 0/0/0
```

The following example shows how to configure a directly attached static route on a broadcast interface.

```
Router(config)# ipv6 route 2001:DB8::1/32 gigabitethernet1/0/0
```

The following example shows how to configure a fully specified static route on a broadcast interface.

```
Router(config)# ipv6 route 2001:DB8::1/32 gigabitethernet1/0/0 fe80::1
```

In the following example, a static route is being configured to a specified next-hop address, from which the output interface is automatically derived.

```
Router(config)# ipv6 route 2001:DB8::/32 2001:DB8:2002:1
```

Sample Output from the show ipv6 static Command When No Options Are Specified in the Command Syntax

When no options are specified in the command, those routes installed in the IPv6 routing table are marked with an asterisk, as shown in the following example:

```
Router# show ipv6 static
IPv6 Static routes
Code: * - installed in RIB
* 2001:DB8:3000:0/16, interface GigabitEthernet1/0/0, distance 1
* 2001:DB8:4000:0/16, via nexthop 2001:DB8:1:1, distance 1
  2001:DB8:5000:0/16, interface GigabitEthernet3/0/0, distance 1
* 2001:DB8:5555:0/16, via nexthop 2001:DB8:4000:1, distance 1
  2001:DB8:5555:0/16, via nexthop 2001:DB8:9999:1, distance 1
* 2001:DB8:5555:0/16, interface GigabitEthernet2/0/0, distance 1
* 2001:DB8:6000:0/16, via nexthop 2001:DB8:2007:1, interface GigabitEthernet1/0/0,
distance 1
```

Sample Output from the show ipv6 static Command with the IPv6 Address and Prefix Command

When the *ipv6-address* or *ipv6-prefix*/*prefix-length* argument is specified, only information about static routes for that address or network is displayed. The following is sample output from the **show ipv6 static**command when entered with the IPv6 prefix 2001:DB8:200::/35:

```
Router# show ipv6 static 2001:DB8:5555:0/16
IPv6 Static routes
Code: * - installed in RIB
* 2001:DB8:5555:0/16, via nexthop 2001:DB8:4000:1, distance 1
  2001:DB8:5555:0/16, via nexthop 2001:9999:1, distance 2
* 2001:DB8:5555:0/16, interface GigabitEthernet2/0/0, distance 1
```

Sample Output from the show ipv6 static interface Command

When an interface is supplied, only those static routes with the specified interface as outgoing interface are displayed. The **interface** keyword may be used with or without the IPv6 address and prefix specified in the **show ipv6 static** command.

```
Router# show ipv6 static interface gigabitethernet3/0/0
IPv6 Static routes
Code: * - installed in RIB
  2001:DB8:5000:)/16, interface GigabitEthernet3/0/0, distance 1
```

Sample Output from the show ipv6 static recursive Command

When the **recursive** keyword is specified in the **show ipv6 static** command, only recursive static routes are displayed. The **recursive** keyword is mutually exclusive with the **interface** keyword, but it may be used >*with* or >*without* the IPv6 prefix included in the command syntax.

```
Router# show ipv6 static recursive
IPv6 Static routes
Code: * - installed in RIB
* 2001:DB8:4000:0/16, via nexthop 2001:DB8:1:1, distance 1
* 2001:DB8:5555:0/16, via nexthop 2001:DB8:4000:1, distance 2
  2001:DB8:5555:0/16, via nexthop 2001:DB8:9999:1, distance 3
```

Sample Output from the show ipv6 static detail Command

When the **detail** keyword is specified, the following additional information is also displayed:

- For >*valid* recursive routes, the output path set, and maximum resolution depth
- For >*invalid* recursive routes, the reason why the route is not valid.
- For >*invalid* direct or fully-specified routes, the reason why the route is not valid.

```
Router# show ipv6 static detail
IPv6 Static routes
Code: * - installed in RIB
* 2001:DB8:3000:0/16, interface GigabitEthernet1/0/0, distance 1
* 2001:DB8:4000:0/16, via nexthop 2001:DB8:2001:1, distance 1
     Resolves to 1 paths (max depth 1)
     via GigabitEthernet1/0/0
  2001:DB8:5000:0/16, interface GigabitEthernet3/0/0, distance 1
     Interface is down
* 2001:DB8:5555:0/16, via nexthop 2001:DB8:4000:1, distance 1
     Resolves to 1 paths (max depth 2)
     via GigabitEthernet1/0/0
  2001:DB8:5555:0/16, via nexthop 2001:DB8:9999:1, distance 1
     Route does not fully resolve
* 2001:DB8:5555:0/16, interface GigabitEthernet2/0/0, distance 1
* 2001:DB8:6000:0/16, via nexthop 2001:DB8:2007:1, interface GigabitEthernet1/0/0,
distance 1
```

Sample Output from the show ipv6 route Command

In the following example, the **show ipv6 route** command is used to verify the configuration of a static route through a point-to-point interface:

```
Router# show ipv6 route
IPv6 Routing Table - 9 entries
Codes: C - Connected, L - Local, S - Static, R - RIP, B - BGP
       U - Per-user Static route
       I1 - ISIS L1, I2 - ISIS L2, IA - ISIS interarea
       O - OSPF intra, OI - OSPF inter, OE1 - OSPF ext 1, OE2 - OSPF ext 2
S   2001:DB8::/32 [1/0]
     via ::, Serial2/0
```

In the following example, the **show ipv6 route** command is used to verify the configuration of a static route on a multiaccess interface. An IPv6 link-local address--FE80::1--is the next-hop router.

```
Router# show ipv6 route
IPv6 Routing Table - 11 entries
Codes: C - Connected, L - Local, S - Static, R - RIP, B - BGP
       U - Per-user Static route
       I1 - ISIS L1, I2 - ISIS L2, IA - ISIS interarea
       O - OSPF intra, OI - OSPF inter, OE1 - OSPF ext 1, OE2 - OSPF ext 2
S   2001:DB8::/32 [1/0]
     via FE80::1, GigabitEthernet0/0/0
```

To display all static routes in the IPv6 routing table, use the **show ipv6 route static** command is used with static as the value of the protocol argument:

```
Router# show ipv6 route static
IPv6 Routing Table - 330 entries
Codes: C - Connected, L - Local, S - Static, R - RIP, B - BGP
       U - Per-user Static route
       I1 - ISIS L1, I2 - ISIS L2, IA - ISIS interarea
S   2001:DB8::/32 [1/0]
     via ::, Tunnel0
S   3FFE:C00:8011::/48 [1/0]
     via ::, Null0
S   ::/0 [254/0]
     via 2001:DB8:2002:806B, Null
```

Sample Output from the debug ipv6 routing Command

In the following example, the **debug ipv6 routing** command is used to verify the installation of a floating static route into the IPv6 routing table when an IPv6 RIP route is deleted. The floating static IPv6 route was previously configured with an administrative distance value of 130. The backup route was added as a floating static route because RIP routes have a default administrative distance of 120, and the RIP route should be the preferred route. When the RIP route is deleted, the floating static route is installed in the IPv6 routing table.

```
Router# debug ipv6 routing
*Oct 10 18:28:00.847: IPv6RT0: rip two, Delete 2001:DB8::/32 from table
*Oct 10 18:28:00.847: IPv6RT0: static, Backup call for 2001:DB8::/32
*Oct 10 18:28:00.847: IPv6RT0: static, Add 2001:DB8::/32 to table
*Oct 10 18:28:00.847: IPv6RT0: static, Adding next-hop :: over Serial2/0 for
2001:DB8::/32, [130/0]
```

Configuration Examples for Implementing Static Routes for IPv6

Static routes may be used for a variety of purposes. Common usages include the following:

- Manual summarization
- Traffic discard
- Fixed default route
- Backup route

In many cases, alternative mechanisms exist within Cisco IOS software to achieve the same objective. Whether to use static routes or one of the alternative mechanisms depends on local circumstances.

- Example Configuring Manual Summarization, page 595
- Example: Configuring Traffic Discard, page 595

- Example: Configuring a Fixed Default Route, page 596
- Example: Configuring a Floating Static Route, page 596

Example Configuring Manual Summarization

The following example shows a static route being used to summarize local interface prefixes advertised into RIP. The static route also serves as a discard route, discarding any packets received by the router to a 2001:DB8:1::/48 destination not covered by a more specific interface prefix.

```
Router> enable
Router# configure terminal
Enter configuration commands, one per line.  End with CNTL/Z.
Router(config)# interface gigabitethernet0/0/0
Router(config-if)# ipv6 address 2001:DB8:2:1234/64
Router(config-if)# exit
Router(config)#
Router(config)# interface gigabitethernet1/0/0
Router(config-if)# ipv6 address 2001:DB8:3:1234/64
Router(config-if)# exit
Router(config)#
Router(config)# interface gigabitethernet2/0/0
Router(config-if)# ipv6 address 2001:DB8:4:1234/64
Router(config-if)# exit
Router(config)#
Router(config)# interface gigabitethernet3/0/0
Router(config-if)# ipv6 address 2001:DB8::1234/64
Router(config-if)# ipv6 rip one enable
Router(config-if)# exit
Router(config)#
Router(config)# ipv6 router rip one
Router(config-rtr)# redistribute static
Router(config-rtr)# exit
Router(config)#
Router(config)# ipv6 route 2001:DB8:1:1/48 null0
Router(config)# end
Router#
00:01:30: %SYS-5-CONFIG_I: Configured from console by console
Router# show ipv6 route static

IPv6 Routing Table - 3 entries
Codes: C - Connected, L - Local, S - Static, R - RIP, B - BGP
       U - Per-user Static route
       I1 - ISIS L1, I2 - ISIS L2, IA - ISIS interarea, IS - ISIS summary
       O - OSPF intra, OI - OSPF inter, OE1 - OSPF ext 1, OE2 - OSPF ext 2
       ON1 - OSPF NSSA ext 1, ON2 - OSPF NSSA ext 2
S   2001:DB8:1::/48 [1/0]
     via ::, Null0
```

Example: Configuring Traffic Discard

Configuring a static route to point at interface null0 may be used for discarding traffic to a particular prefix. For example, if it is required to discard all traffic to prefix 2001:DB8:42:1/64, the following static route would be defined:

```
Device> enable
Device# configure
 terminal
Enter configuration commands, one per line.  End with CNTL/Z.
Device(config)# ipv6 route 2001:DB8:42:1::/64 null0
Device(config)# end
```

Example: Configuring a Fixed Default Route

A default static route is often used in simple router topologies. In the following example, a router is connected to its local site via GigabitEthernet 0/0/0 and to the main corporate network via Serial 2/0/0 and Serial 3/0/0. All nonlocal traffic will be routed over the two serial interfaces.

```
Router(config)# interface gigabitethernet0/0/0
Router(config-if)# ipv6 address 2001:DB8:17:1234/64
Router(config-if)# exit
Router(config)# interface Serial2/0/0
Router(config-if)# ipv6 address 2001:DB8:1:1234/64
Router(config-if)# exit
Router(config)# interface Serial3/0/0
Router(config-if)# ipv6 address 2001:DB8:2:124/64
Router(config-if)# exit
Router(config)# ipv6 route ::/0 Serial2/0
Router(config)# ipv6 route ::/0 Serial3/0
Router(config)# end
Router#
00:06:30: %SYS-5-CONFIG_I: Configured from console by console
Router# show ipv6 route static
IPv6 Routing Table - 7 entries
Codes: C - Connected, L - Local, S - Static, R - RIP, B - BGP
       U - Per-user Static route
       I1 - ISIS L1, I2 - ISIS L2, IA - ISIS interarea, IS - ISIS summary
       O - OSPF intra, OI - OSPF inter, OE1 - OSPF ext 1, OE2 - OSPF ext 2
       ON1 - OSPF NSSA ext 1, ON2 - OSPF NSSA ext 2
S   ::/0 [1/0]
     via ::, Serial2/0
     via ::, Serial3/0
```

Example: Configuring a Floating Static Route

A floating static route often is used to provide a backup path in the event of connectivity failure. In the following example, the router has connectivity to the network core via GigabitEthernet0/0/0 and learns the route 2001:DB8:1:1/32 via IS-IS. If the GigabitEthernet0/0/0 interface fails, or if route 2001:DB8:1:1/32 is no longer learned via IS-IS (indicating loss of connectivity elsewhere in the network), traffic is routed via the backup ISDN interface.

```
Router> enable
Router# configure
 terminal
Enter configuration commands, one per line.  End with CNTL/Z.
Router(config)# interface gigabitethernet0/0/0
Router(config-if)# ipv6 address 2001:DB8:17:1234/64
Router(config-if)# exit
Router(config)# interface gigabitethernet0/0/0
Router(config-if)# ipv6 address 2001:DB8:1:1234/64
Router(config-if)# ipv6
 router
 isis
Router(config-if)# exit
Router(config)# router isis
Router(config-router)# net 42.0000.0000.0000.0001.00
Router(config-router)# exit
Router(config)# interface BRI1/0
Router(config-if)# encapsulation ppp
Router(config-if)# ipv6 enable
Router(config-if)# isdn switch-type basic-net3
Router(config-if)# ppp authentication chap optional
Router(config-if)# ppp multilink
Router(config-if)# exit
Router(config)# dialer-list 1 protocol ipv6 permit
Router(config)# ipv6 route 2001:DB8:1::/32 BRI1/0 200
Router(config)# end
```

```
Router#
00:03:07: %SYS-5-CONFIG_I: Configured from console by console
```

Additional References

Related Documents

Related Topic	Document Title
IP static route configuration	" Configuring IP Routing Protocol-Independent Features," *Cisco IOS XE IP Routing Protocols Configuration Guide* , Release 2
IP static route commands: complete command syntax, command mode, defaults, usage guidelines, and examples	*Cisco IOS IP Routing Protocols Command Reference*
IPv6 supported feature list	"Start Here: Cisco IOS XE Software Release Specifics for IPv6 Features ," *Cisco IOS XE IPv6 Configuration Guide, Release 2*
IPv6 commands: complete command syntax, command mode, defaults, usage guidelines, and examples	*Cisco IOS IPv6 Command Reference*
Cisco IOS master command list, all releases	Cisco IOS Master Command List, All Releases

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
None	To locate and download MIBs for selected platforms, Cisco software releases, and feature sets, use Cisco MIB Locator found at the following URL: http://www.cisco.com/go/mibs

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

Feature Information for Implementing Static Routes for IPv6

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Table 33 Feature Information for Implementing Static Routes for IPv6

Feature Name	Releases	Feature Information
IPv6 Routing--Static Routing	Cisco IOS XE Release 2.1	Static routes are manually configured and define an explicit path between two networking devices. The following commands were modified by this feature: **debug ipv6 route**, **ipv6 route**, **ipv6 route static bfd**, **monitor event ipv6 static**, **show ipv6 route**, **show ipv6 route summary**, **show ipv6 static**

Implementing Traffic Filters for IPv6 Security

This module describes how to configure Cisco IOS XE IPv6 traffic filter and firewall features for your Cisco networking devices. These security features can protect your network from degradation or failure and also from data loss or compromised security resulting from intentional attacks and from unintended but damaging mistakes by well-meaning network users.

Finding Feature Information

Your software release may not support all the features documented in this module. For the latest caveats and feature information, see Bug Search Tool and the release notes for your platform and software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the feature information table at the end of this module.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Restrictions for Implementing Traffic Filters for IPv6 Security

- In Cisco IOS XE software, the standard IPv6 access control list (ACL) functionality is extended to support traffic filtering based on IPv6 option headers and optional, upper-layer protocol type information for finer granularity of control (functionality similar to extended ACLs in IPv4).
- The IPv6 Template ACL feature applies only to virtual access interfaces and sessions with ACLs defined using RADIUS. ACLs on vty interfaces or named ACLs on physical interfaces are not supported by this feature.
- The IPv6 Template ACL feature supports vendor-specific attribute (VSA) Cisco AV-pairs only. It does not support the Attribute 242 ACL.

Information About Implementing Traffic Filters for IPv6 Security

Access Control Lists for IPv6 Traffic Filtering

The standard ACL functionality in IPv6 is similar to standard ACLs in IPv4. Access lists determine what traffic is blocked and what traffic is forwarded at router interfaces and allow filtering based on source and destination addresses, inbound and outbound to a specific interface. Each access list has an implicit deny statement at the end. IPv6 ACLs are defined and their deny and permit conditions are set using the **ipv6 access-list** command with the **deny** and **permit** keywords in global configuration mode.

Named and tagged ACLs are both supported in IPv6:

- A named ACL consists of one or more access control entry (ACE) and is defined on the Intelligent Service Gateway (ISG) router by name.
- A name for a tagged ACL is dynamically created by the AAA when the ACL is applied. These ACEs are defined on the RADIUS.

IPv6 Packet Inspection

The following header fields are all used for IPv6 inspection--traffic class, flow label, payload length, next header, hop limit, and source or destination address. For further information on and descriptions of the IPv6 header fields, see RFC 2474.

Tunneling Support

IPv6 packets tunneled in IPv4 are not inspected. If a tunnel terminates on a router, and IPv6 traffic exiting the tunnel is nonterminating, then the traffic is inspected.

Virtual Fragmentation Reassembly

When virtual fragmentation reassembly (VFR) is enabled, VFR processing begins after ACL input lists are checked against incoming packets. The incoming packets are tagged with the appropriate VFR information.

Access Class Filtering in IPv6

Filtering incoming and outgoing connections to and from the router based on an IPv6 ACL is performed using the **ipv6 access-class** command in line configuration mode. The **ipv6 access-class** command is similar to the **access-class** command, except the IPv6 ACLs are defined by a name. If the IPv6 ACL is

applied to inbound traffic, the source address in the ACL is matched against the incoming connection source address and the destination address in the ACL is matched against the local router address on the interface. If the IPv6 ACL is applied to outbound traffic, the source address in the ACL is matched against the local router address on the interface and the destination address in the ACL is matched against the outgoing connection source address. We recommend that identical restrictions are set on all the virtual terminal lines because a user can attempt to connect to any of them.

IPv6 Template ACL

When user profiles are configured using vendor-specific attribute (VSA) Cisco AV-pairs, similar per-user IPv6 ACLs may be replaced by a single template ACL. That is, one ACL represents many similar ACLs. By using IPv6 template ACLs, you can increase the total number of per-user ACLs while minimizing the memory and Ternary Content Addressable Memory (TCAM) resources needed to support the ACLs.

The IPv6 Template ACL feature can create templates using the following ACL fields:

- IPv6 source and destination addresses
- TCP and UDP, including all associated ports (0 through 65535)
- ICMP neighbor discovery advertisements and solicitations
- IPv6 DSCP with specified DSCP values

ACL names are dynamically generated by this feature; for example:

- 6Temp_#152875854573--Example of a dynamically generated template name for a template ACL parent
- Virtual-Access2.32135#152875854573--Example of a child ACL or an ACL that has not yet been made part of a template.

SSO ISSU Support for Per-User IPv6 ACL for PPP Sessions

The SSO/ISSU Support for per-User IPv6 ACL for PPP Sessions feature reproduces IPv6 ACLs on the active Route Processor to the standby RP and provides a consistent stateful switchover and in-service software upgrade experience for active sessions. The feature also extends the ability to maintain Template ACLs (IPv6 only or dual stack) through ISSU and SSO.

Both named and tagged ACLs can be configured and applied in the following ways:

- Virtual-template ACL:
    - Virtual-template ACLs (also called interface ACLs) are configured under a virtual-template definition on the ISG router.
    - Only named ACLs can be configured under a virtual-template definition. Named ACLs applied to virtual templates get cloned to all virtual access interfaces created using that virtual-template definition.
- Per-user ACLs are always applied through RADIUS:
    - User profile--The ACL is configured in the user profile on RADIUS and is applied when the session is up.
    - Change of Authorization (CoA) per-user push--The ACL is applied through a RADIUS CoA push from a subscriber profile.

The table below shows information about support for functionality and SSO for these ACL configurations:

Table 34 SSO Support for Named and Tagged ACLs

ACL Configuration	Functionality Supported	SSO Supported
Named ACL		
Virtual-Template	Yes	Yes
User Profile	Yes	Yes
CoA per-User Push	Yes	No
Tagged ACL		
Virtual-Template	No	No
User Profile	Yes	Yes
CoA per-User Push	Yes	No

How to Implement Traffic Filters for IPv6 Security

Configuring IPv6 Traffic Filtering

Creating and Configuring an IPv6 ACL for Traffic Filtering

Note IPv6 ACLs on the Cisco ASR 1000 platform do not contain implicit permit rules. The IPv6 neighbor discovery process uses the IPv6 network-layer service; therefore, to enable IPv6 neighbor discovery, you must add IPv6 ACLs to allow IPv6 neighbor discovery packets to be sent and received on an interface. In IPv4, the Address Resolution Protocol (ARP), which is equivalent to the IPv6 neighbor discovery process, uses a separate data-link-layer protocol; therefore, by default IPv4 ACLs implicitly allow ARP packets to be sent and received on an interface.

>

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ipv6 access-list** *access-list-name*
4. Do one of the following:
   - **permit protocol** {*source-ipv6-prefix/prefix-length* | **any** | **host** *source-ipv6-address*} [*operator* [*port-number*]] {*destination-ipv6-prefix / prefix-length* | **any** | **host** *destination-ipv6-address*} [**operator** [*port-number*]] [**dest-option-type** [*doh-number* | *doh-type*]] [**dscp** *value*] [**flow-label** *value*] [**fragments**] [**log**] [**log-input**] [**mobility**] [**mobility-type** [*mh-number* | *mh-type*]] [**routing**] [**routing-type** *routing-number*] [**sequence** *value*] [**time-range** *name*]
   -
   -
   - **deny** *protocol* {*source-ipv6-prefix/prefix-length* | **any** | **host** *source-ipv6-address*} [*operator port-number*]] {*destination-ipv6-prefix/prefix-length* | **any** | **host** *destination-ipv6-address*} [*operator* [*port-number*]] [**dest-option-type** [*doh-number* | *doh-type*]] [**dscp** *value*] [**flow-label** *value*] [**fragments**] [**log**] [**log-input**] [**mobility**] [**mobility-type** [*mh-number* | *mh-type*]] [**routing**] [**routing-type** *routing-number*] [**sequence** *value*] [**time-range** *name*] [**undetermined-transport**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**ipv6 access-list** *access-list-name* **Example:** `Router(config)# ipv6 access-list outbound`	Defines an IPv6 ACL, and enters IPv6 access list configuration mode. • The *access-list name*argument specifies the name of the IPv6 ACL. IPv6 ACL names cannot contain a space or quotation mark, or begin with a numeral.

Command or Action	Purpose

Step 4 Do one of the following:
• **permit protocol** {*source-ipv6-prefix/prefix-length* | **any** | **host** *source-ipv6-address*} [*operator* [*port-number*]] {*destination-ipv6-prefix / prefix-length* | **any** | **host** *destination-ipv6-address*} [**operator** [*port-number*]] [**dest-option-type** [*doh-number* | *doh-type*]] [**dscp** *value*] [**flow-label** *value*] [**fragments**] [**log**] [**log-input**] [**mobility**] [**mobility-type** [*mh-number* | *mh-type*]] [**routing**] [**routing-type** *routing-number*] [**sequence** *value*] [**time-range** *name*]
•
•
• **deny** *protocol* {*source-ipv6-prefix/prefix-length* | **any** | **host** *source-ipv6-address*} [*operator port-number*]] {*destination-ipv6-prefix/ prefix-length* | **any** | **host** *destination-ipv6-address*} [*operator* [*port-number*]] [**dest-option-type** [*doh-number* | *doh-type*]] [**dscp** *value*] [**flow-label** *value*] [**fragments**] [**log**] [**log-input**] [**mobility**] [**mobility-type** [*mh-number* | *mh-type*]] [**routing**] [**routing-type** *routing-number*] [**sequence** *value*] [**time-range** *name*] [**undetermined-transport**
Example:
`Router(config-ipv6-acl)# permit tcp 2001:DB8:0300:0201::/32 eq telnet any`
Example:
Example:
Example:
`Router(config-ipv6-acl)# deny tcp host 2001:DB8:1::1 any log-input`
	Specifies permit or deny conditions for an IPv6 ACL.

Applying the IPv6 ACL to an Interface

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type number*
4. **ipv6 traffic-filter** *access-list-name* {**in**| **out**}

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**interface** *type number* **Example:** `Router(config)# interface gigabitethernet 0/0/0`	Specifies the interface type and number, and enters interface configuration mode.
Step 4	**ipv6 traffic-filter** *access-list-name* {**in**| **out**} **Example:** `Router(config-if)# ipv6 traffic-filter outbound out`	Applies the specified IPv6 access list to the interface specified in the previous step.

Controlling Access to a vty

Creating an IPv6 ACL to Provide Access Class Filtering

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ipv6 access-list** *access-list-name*
4. Do one of the following:
   - **permit protocol** {*source-ipv6-prefix/prefix-length* | **any** | **host** *source-ipv6-address*} [*operator* [*port-number*]] {*destination-ipv6-prefix / prefix-length* | **any** | **host** *destination-ipv6-address*} [*operator* [*port-number*]] [**dest-option-type** [*doh-number* | *doh-type*]] [**dscp** *value*] [**flow-label** *value*] [**fragments**] [**log**] [**log-input**] [**mobility**] [**mobility-type** [*mh-number* | *mh-type*]] [**routing**] [**routing-type** *routing-number*] [**sequence** *value*] [**time-range** *name*
   -
   -
   - **deny** *protocol* {*source-ipv6-prefix*/*prefix-length* | **any** | **host** *source-ipv6-address*} [*operator port-number*]] {*destination-ipv6-prefix/prefix-length* | **any** | **host** *destination-ipv6-address*} [*operator* [*port-number*]] [**dest-option-type** [*doh-number* | *doh-type*]] [**dscp** *value*] [**flow-label** *value*] [**fragments**] [**log**] [**log-input**] [**mobility**] [**mobility-type** [*mh-number* | *mh-type*]] [**routing**] [**routing-type** *routing-number*] [**sequence** *value*] [**time-range** *name*] [**undetermined-transport**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**ipv6 access-list** *access-list-name* **Example:** `Router(config)# ipv6 access-list cisco`	Defines an IPv6 ACL, and enters IPv6 access list configuration mode.

Command or Action	Purpose

Step 4 Do one of the following:
• **permit protocol** {*source-ipv6-prefix/prefix-length* | **any** | **host** *source-ipv6-address*} [*operator* [*port-number*]] {*destination-ipv6-prefix / prefix-length* | **any** | **host** *destination-ipv6-address*} [*operator* [*port-number*]] [**dest-option-type** [*doh-number* | *doh-type*]] [**dscp** *value*] [**flow-label** *value*] [**fragments**] [**log**] [**log-input**] [**mobility**] [**mobility-type** [*mh-number* | *mh-type*]] [**routing**] [**routing-type** *routing-number*] [**sequence** *value*] [**time-range** *name*
•
•
• **deny** *protocol* {*source-ipv6-prefix/prefix-length* | **any** | **host** *source-ipv6-address*} [*operator port-number*]] {*destination-ipv6-prefix/prefix-length* | **any** | **host** *destination-ipv6-address*} [*operator* [*port-number*]] [**dest-option-type** [*doh-number* | *doh-type*]] [**dscp** *value*] [**flow-label** *value*] [**fragments**] [**log**] [**log-input**] [**mobility**] [**mobility-type** [*mh-number* | *mh-type*]] [**routing**] [**routing-type** *routing-number*] [**sequence** *value*] [**time-range** *name*] [**undetermined-transport**
Example:
`Router(config-ipv6-acl)# permit ipv6 host 2001:DB8:0:4::32 any eq telnet`
Example:
Example:
Example:
`Router(config-ipv6-acl)# deny ipv6 host 2001:DB8:0:6::6/32 any`
	Specifies permit or deny conditions for an IPv6 ACL.

Applying an IPv6 ACL to the Virtual Terminal Line

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **line** [**aux**| **console**| **tty**| **vty**] *line-number*[*ending-line-number*]
4. **ipv6 access-class** *ipv6-access-list-name* {**in**| **out**}

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**line** [**aux**| **console**| **tty**| **vty**] *line-number*[*ending-line-number*] **Example:** `Router(config)# line vty 0 4`	Identifies a specific line for configuration and enters line configuration mode. • In this example, the **vty** keyword is used to specify the virtual terminal lines for remote console access.
Step 4	**ipv6 access-class** *ipv6-access-list-name* {**in**| **out**} **Example:** `Router(config-line)# ipv6 access-class cisco in`	Filters incoming and outgoing connections to and from the router based on an IPv6 ACL.

Enabling IPv6 Template Processing

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **access-list template** [*number-of-rules*]
4. **exit**
5. **show access-list template** {**summary** | *aclname* | **exceed** *number* | **tree**}

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**access-list template** [*number-of-rules*] **Example:** `Router(config)# access-list template 50`	Enables template ACL processing. • The example in this task specifies that ACLs with 50 or fewer rules will be considered for template ACL status. • The *number-of-rules* argument default is 100.
Step 4	**exit** **Example:** `Router(config)# exit`	Exits global configuration mode and places the router in privileged EXEC mode.
Step 5	**show access-list template** {**summary** | *aclname* | **exceed** *number* | **tree**} **Example:** `Router# show access-list template summary`	Displays information about ACL templates.

Troubleshooting IPv6 Security Configuration and Operation

SUMMARY STEPS

1. **enable**
2. **clear ipv6 access-list** [*access-list-name*]
3. **clea r ipv6 inspect** {**session** *session-number* | **all**}
4. **clear ipv6 prefix-list** [*prefix-list-name*] [*ipv6-prefix*/*prefix-length*
5. **debug platform software acl config**
6. **debug platform software acl interface**
7. **debug platform software acl statistics**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router# enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**clear ipv6 access-list** [*access-list-name*] **Example:** `Router# clear ipv6 access-list list1`	Resets the IPv6 access list match counters.
Step 3	**clea r ipv6 inspect** {**session** *session-number* | **all**} **Example:** `Router# clear ipv6 inspect all`	Removes a specific IPv6 session or all IPv6 inspection sessions.
Step 4	**clear ipv6 prefix-list** [*prefix-list-name*] [*ipv6-prefix*/*prefix-length* **Example:** `Router# clear ipv6 prefix-list`	Resets the hit count of the IPv6 prefix list entries.
Step 5	**debug platform software acl config** **Example:** `Router# debug platform software acl config`	Enables debugging for ACL configuration changes, such as addition, deletion, or editing of an ACL and ACL entries.
Step 6	**debug platform software acl interface** **Example:** `Router# debug platform software acl interface`	Enables debugging for interface ACL configurations, such as applying or removing an ACL to or from an interface.
Step 7	**debug platform software acl statistics** **Example:** `Router# debug platform software acl statistics`	Enables statistics update messages from the Forwarding Processor Forwarding Manager.

Configuration Examples for Implementing Traffic Filters for IPv6 Security

Example Configuring an Access List on the Router

Example: Route Processor Forwarding Manager ACL Configuration

```
Router# show running-config interface port-channel 3.2
Building configuration...
Current configuration : 328 bytes
!
interface Port-channel3.2
 encapsulation dot1Q 2 primary GigabitEthernet0/0/4 secondary GigabitEthernet1/2/4
 ip address 10.1.1.1 255.255.255.0
 ipv6 address 2001:DB8:1111:1111::1/64
 ipv6 traffic-filter OutFilter_IPv6 out
 ipv6 nd reachable-time 180000
 ipv6 nd ra suppress
 ipv6 ospf 100 area 0
 snmp trap link-status
end
Router# show ipv6 access-list OutFilter_IPv6
IPv6 access list OutFilter_IPv6
    permit icmp any any mld-query sequence 30
    permit icmp any any router-advertisement sequence 40
    deny 103 any any sequence 50
    permit icmp any any packet-too-big sequence 60
    deny icmp any any sequence 70
    deny ipv6 2404:1A8:1100:9::/64 any sequence 74
    deny ipv6 2404:1A8:1100:10::/64 any sequence 75
    permit ipv6 any 2050::/16 log-input sequence 80
    deny ipv6 2404:1A8:1100:13::/64 any sequence 90
    deny ipv6 2404:1A8:1100:14::/64 any sequence 100
    deny ipv6 2408:40:2000::/35 2408:40:2000::/35 dscp default sequence 110
    permit ipv6 any any (3974749339 matches) sequence 120
Router# show platform software access-list R0 statistics
Forwarding Manager Access-list Messaging Statistics
Set Log Threshold: 0, Interval: 0
IPv4 Access-list Entry Add: 1, Delete: 0
IPv4 Access-list Bind: 0, Unbind: 0
IPv4 Access-list Resequence: 0, Delete: 1
IPv6 Access-list Entry Add: 82, Delete: 0
IPv6 Access-list Bind: 3003, Unbind: 0
IPv6 Access-list Resequence: 0, Delete: 0
Access-list Sync Start: 0, End: 0
CPP Match Add: 0, Replace: 0, ACK Success: 0, ACK Error: 0
CPP Match Delete: 0, ACK Success: 0, ACK Error: 0
CPP Action Edit: 0, ACK Success: 0, ACK Error: 0
CPP Action Replace: 0, ACK Success: 0, ACK Error: 0
CPP Bind: 0, ACK Success: 0, ACK Error: 0
CPP Unbind: 0, ACK Success: 0, ACK Error: 0
Router# show platform software access-list R1 name OutFilter_IPv6 ace 100
```

```
Access-list: OutFilter_IPv6
Access-list Entry Sequence: 100
  Type: Permanent, Operation: Add
  Action: Deny
  Destination Address: ::, Length: 00
  Source Address: 2404:1a8:1100:14::, Length: 0x24
```

Example: Forwarding Processor Forwarding Manager ACL Configuration

```
Router# show platform software access-list F0 statistics
Forwarding Manager Access-list Messaging Statistics
Set Log Threshold: 0, Interval: 0
IPv4 Access-list Entry Add: 0, Delete: 0
IPv4 Access-list Bind: 0, Unbind: 0
IPv4 Access-list Resequence: 0, Delete: 1
IPv6 Access-list Entry Add: 82, Delete: 0
IPv6 Access-list Bind: 3003, Unbind: 0
IPv6 Access-list Resequence: 0, Delete: 0
Access-list Sync Start: 0, End: 0
CPP Match Add: 86, Replace: 0, ACK Success: 86, ACK Error: 0
CPP Match Delete: 4, ACK Success: 4, ACK Error: 0
CPP Action Edit: 83, ACK Success: 83, ACK Error: 0
CPP Action Replace: 0, ACK Success: 0, ACK Error: 0
CPP Bind: 3003, ACK Success: 3003, ACK Error: 0
CPP Unbind: 0, ACK Success: 0, ACK Error: 0
Router# show platform software access-list F0 name OutFilter_IPv6 ace 100
  Access-list: OutFilter_IPv6
  Access-list Entry Sequence: 100
  Match Class Index: 11
  Epoch: 0
  State: Downloaded
  Requested Operation: No-op
  Issued Operation: No-op
  Type: Permanent
Action: Deny
Router# access-list F0 name OutFilter_IPv6 ace 100 max-records 20
Access-list: OutFilter_IPv6
Access-list Index: 2, Protocol: IPv6, Type: IPv6
  Security References: 2001, Classifier References: 0, Shared target: 2001
  Pending Download Access-list Entry: 0
  Pending Acknowledgements Matches: 0, Actions: 0
  Downloaded Access-list Entry: 12
  Total Access-list Entry after pending updates are processed: 12
  AOM object identifier: 141
  State: Normal
  Number of Access-list Entry Shown: 3
  ACE Number  Class Index  State
  ------------------------------------------------------------------
  100         11           Downloaded
  110         12           Downloaded
  120         13           Downloaded
```

The following command summarizes the number of entries and references in the access list:

```
Router# show platform software access-list F0 summary
Access-list                              Index        Num Ref      Num ACEs
-------------------------------------------------------------------------------
icmp2                                    1            1            2
OutFilter_IPv6                           2            2001         12
p11                                      3            1000         3
```

Example Applying an IPv6 Access List to an Interface

- Example: Applying the Route Processor Forwarding Manager ACL to an Interface, page 615
- Example: Applying the Forwarding Processor Forwarding Manager ACL to an Interface, page 615

Example: Applying the Route Processor Forwarding Manager ACL to an Interface

The following examples show how to configure and verify the Route Processor Forwarding Manager access list application to Gigabit Ethernet interface 1/0/1:

```
Router(config)# interface GigabitEthernet 1/0/1
Router(config-if)# ip access-group test in
Router# show platform software access-list R0 statistics
Forwarding Manager Access-list Messaging Statistics
Set Log Threshold: 0, Interval: 0
IPv4 Access-list Entry Add: 1, Delete: 0
IPv4 Access-list Bind: 0, Unbind: 0
IPv4 Access-list Resequence: 0, Delete: 1
IPv6 Access-list Entry Add: 82, Delete: 0
IPv6 Access-list Bind: 3003, Unbind: 0
IPv6 Access-list Resequence: 0, Delete: 0
Access-list Sync Start: 0, End: 0
CPP Match Add: 0, Replace: 0, ACK Success: 0, ACK Error: 0
CPP Match Delete: 0, ACK Success: 0, ACK Error: 0
CPP Action Edit: 0, ACK Success: 0, ACK Error: 0
CPP Action Replace: 0, ACK Success: 0, ACK Error: 0
CPP Bind: 0, ACK Success: 0, ACK Error: 0
CPP Unbind: 0, ACK Success: 0, ACK Error: 0
Router# show platform software access-list R0 bind interface Port-channel1.2
Interface: Port-channel1.2, Index: 35, Protocol: IPv6, Direction: Output
  Access-list: OutFilter_IPv6
  Operation: Add
```

Example: Applying the Forwarding Processor Forwarding Manager ACL to an Interface

The following examples show how to configure and verify the Forwarding Processor Forwarding Manager access list application to Gigabit Ethernet interface 1/0/1:

```
Router(config)# interface GigabitEthernet 1/0/1
Router(config-if)# ip access-group test in
Router# show platform software access-list F0 statistics

Forwarding Manager Access-list Messaging Statistics
Set Log Threshold: 0, Interval: 0
IPv4 Access-list Entry Add: 0, Delete: 0
IPv4 Access-list Bind: 0, Unbind: 0
IPv4 Access-list Resequence: 0, Delete: 1
IPv6 Access-list Entry Add: 82, Delete: 0
IPv6 Access-list Bind: 3003, Unbind: 0
IPv6 Access-list Resequence: 0, Delete: 0
Access-list Sync Start: 0, End: 0
CPP Match Add: 86, Replace: 0, ACK Success: 86, ACK Error: 0
CPP Match Delete: 4, ACK Success: 4, ACK Error: 0
CPP Action Edit: 83, ACK Success: 83, ACK Error: 0
CPP Action Replace: 0, ACK Success: 0, ACK Error: 0
CPP Bind: 3003, ACK Success: 3003, ACK Error: 0
CPP Unbind: 0, ACK Success: 0, ACK Error: 0
```

The following example provides a summary of the access list with number of entries and number of references:

```
Router# show platform software access-list F0 summary
Access-list                          Index        Num Ref      Num ACEs
-------------------------------------------------------------------------
icmp2                                  1            1             2
OutFilter_IPv6                         2            2001         12
p11                                    3            1000          3
m1                                     4            1             2
p1                                     5            0             3
```

Example: IPv6 Template ACL Processing

In this example, the contents of ACL1 and ACL2 are the same, but the names are different:

```
ipv6 access-list extended ACL1 (PeerIP: 2001:1::1/64)
permit igmp any                 2003:1::1/64
permit icmp 2002:5::B/64        any
permit udp  any                 host 2004:1::5
permit udp  any                 host 2002:2BC::a
permit icmp host 2001:BC::7     host 2003:3::7
ipv6 access-list extended ACL2 (PeerIP: 2007:2::7/64)
permit igmp any                 2003:1::1/64
permit icmp 2002:5::B/64        any
permit udp  any                 host 2004:1::5
permit udp  any                 host 2002:2BC::a
permit icmp host 2001:BC::7     host 2003:3::7
```

The template for these ACLs is as follows:

```
ipv6 access-list extended Template_1
permit igmp any                 2003:1::1/64
permit icmp 2002:5::B/64        any
permit udp  any                 host 2004:1::5
permit udp  any                 host 2002:2BC::a
permit icmp host 2001:BC::7     host 2003:3::7
```

Example Displaying Access List Statistics

The following example output for ACL statistics provides information about the counter aggregation and poll timer:

```
Router# show ipv6 access-list OutFilter_IPv6
IPv6 access list OutFilter_IPv6
    permit icmp any any mld-query sequence 30
    permit icmp any any router-advertisement sequence 40
    deny 103 any any sequence 50
    permit icmp any any packet-too-big sequence 60
    deny icmp any any sequence 70
    deny ipv6 2001:DB8:1100:9::/64 any sequence 74
    deny ipv6 2001:DB8:1100:10::/64 any sequence 75
    permit ipv6 any 2050::/16 log-input sequence 80
    deny ipv6 2001:DB8:1100:13::/64 any sequence 90
    deny ipv6 2001:DB8:1100:14::/64 any sequence 100
    deny ipv6 2001:DB8:2000::/35 2408:40:2000::/35 dscp default sequence 110
    permit ipv6 any any (175392444 matches) sequence 120
```

Additional References

Related Documents

Related Topic	Document Title
Basic IPv6 configuration	" Implementing IPv6 Addressing and Basic Connectivity ," *Cisco IOS XE IPv6 Configuration Guide*

Related Topic	Document Title
IPv6 supported feature list	" Start Here: Cisco IOS XE Software Release Specifics for IPv6 Features ," *Cisco IOS XE IPv6 Configuration Guide*
Stateful Switchover	Configuring Stateful Switchover
In Service Software Upgrade	Cisco IOS XE In Service Software Upgrade Process
IPv6 commands: complete command syntax, command mode, defaults, usage guidelines, and examples	*Cisco IOS IPv6 Command Reference*
Cisco IOS master command list, all releases	Cisco IOS Master Command List, All Releases

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
None	To locate and download MIBs for selected platforms, Cisco software releases, and feature sets, use Cisco MIB Locator found at the following URL: http://www.cisco.com/go/mibs

RFCs

RFCs	Title
RFC 2401	*Security Architecture for the Internet Protocol*
RFC 2402	*IP Authentication Header*
RFC 2428	*FTP Extensions for IPv6 and NATs*
RFC 2460	*Internet Protocol, Version 6 (IPv6) Specification*
RFC 2474	*Definition of the Differentiated Services Field (DS Field) in the IPv4 and IPv6 Headers*
RFC 3576	*Change of Authorization*

RFCs	Title
RFC 4241	*A Model of IPv6/IPv4 Dual Stack Internet Access Service*

Technical Assistance

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

Feature Information for Implementing Traffic Filters for IPv6 Security

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Table 35 Feature Information for Implementing Traffic Filters for IPv6 Security

Feature Name	Releases	Feature Information
IPv6 Services--Extended Access Control Lists	Cisco IOS XE Release 2.1	Standard IPv6 ACL functionality was extended to support traffic filtering based on IPv6 option headers and optional, upper-layer protocol type information for finer granularity of control. The following commands were modified by this feature: **clear ipv6 access-list**, **clear ipv6 inspect**, **clear ipv6 prefix-list**, **deny**, **ipv6 access-class**, **ipv6 access-list**, **ipv6 traffic-filter**, **line**, **permit**, **show ipv6 access-list**.

Feature Name	Releases	Feature Information
IPv6 Services--Standard Access Control Lists	Cisco IOS XE Release 2.1	Access lists determine what traffic is blocked and what traffic is forwarded at router interfaces and allow filtering based on source and destination addresses, inbound and outbound to a specific interface. The following commands were modified by this feature: **clear ipv6 access-list**, **clear ipv6 inspect**, **clear ipv6 prefix-list**, **deny**, **ipv6 access-class**, **ipv6 access-list**, **ipv6 traffic-filter**, **line**, **permit**, **show ipv6 access-list**.
IPv6 ACL--Template ACL	Cisco IOS XE Release 3.2S	This feature allows similar per-user IPv6 ACLs to be replaced by a single template ACL. The following commands were modified by this feature: **access-list template**, **show access-list template**.
SSO/ISSU Support for Per-User IPv6 ACL for PPP Sessions	Cisco IOS XE Release 3.2.1S	Reproducing IPv6 ACLs on the active RP to the standby RP provides a consistent SSO and ISSU experience for active sessions. The following section provides information about this feature:

IPv6 ACL Extensions for Hop by Hop Filtering

The IPv6 ACL Extensions for Hop by Hop Filtering feature allows you to control IPv6 traffic that might contain hop-by-hop extension headers. You can configure an access-control list (ACL) to deny all hop-by-hop traffic or to selectively permit traffic based on protocol.

Finding Feature Information

Your software release may not support all the features documented in this module. For the latest caveats and feature information, see Bug Search Tool and the release notes for your platform and software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the feature information table at the end of this module.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Information About IPv6 ACL Extensions for Hop by Hop Filtering

ACLs and Traffic Forwarding

IPv6 ACLs determine what traffic is blocked and what traffic is forwarded at device interfaces. ACLs allow filtering based on source and destination addresses, inbound and outbound to a specific interface. Use the **ipv6 access-list** command to define an IPv6 ACL, and the **deny (ipv6)** and **permit (ipv6)** commands to configure its conditions.

The IPv6 ACL Extensions for Hop by Hop Filtering feature implements RFC 2460 to support traffic filtering in any upper-layer protocol type.

How to Configure IPv6 ACL Extensions for Hop by Hop Filtering

Configuring IPv6 ACL Extensions for Hop by Hop Filtering

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ipv6 access-list** *access-list-name*
4. **permit** *protocol* {*source-ipv6-prefix/prefix-length* | **any** | **host** *source-ipv6-address* | **auth**} [*operator* [*port-number*]] {*destination-ipv6-prefix/prefix-length* | **any** | **host** *destination-ipv6-address*| **auth**} [*operator* [*port-number*]] [**dest-option-type** [*doh-number* | *doh-type*]] [**dscp** *value*] [**flow-label** *value*] [**fragments**] [**hbh**] [**log**] [**log-input**] [**mobility**] [**mobility-type** [*mh-number* | *mh-type*]] [**reflect** *name* [**timeout** *value*]] [**routing**] [**routing-type** *routing-number*] [**sequence** *value*] [**time-range** *name*]
5. **deny** *protocol* {*source-ipv6-prefix/prefix-length* | **any**| **host** *source-ipv6-address* | **auth**} [*operator*[*port-number*]] {*destination-ipv6-prefix/prefix-length* | **any** | **host** *destination-ipv6-address* | **auth**} [*operator* [*port-number*]] [**dest-option-type** [*doh-number* | *doh-type*]] [**dscp** *value*] [**flow-label** *value*] [**fragments**] [**hbh**] [**log**] [**log-input**] [**mobility**] [**mobility-type** [*mh-number* | *mh-type*]] [**routing**] [**routing-type** *routing-number*] [**sequence** *value*] [**time-range** *name*] [**undetermined-transport**]
6. **end**

DETAILED STEPS

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

	Command or Action	Purpose
Step 3	**ipv6 access-list** *access-list-name* **Example:** `Device(config)# ipv6 access-list hbh-acl`	Defines an IPv6 ACL, and enters IPv6 access list configuration mode.
Step 4	**permit** *protocol* {*source-ipv6-prefix/prefix-length* | **any** | **host** *source-ipv6-address* | **auth**} [*operator* [*port-number*]] {*destination-ipv6-prefix/prefix-length* | **any** | **host** *destination-ipv6-address*| **auth**} [*operator* [*port-number*]] [**dest-option-type** [*doh-number* | *doh-type*]] [**dscp** *value*] [**flow-label** *value*] [**fragments**] [**hbh**] [**log**] [**log-input**] [**mobility**] [**mobility-type** [*mh-number* | *mh-type*]] [**reflect** *name* [**timeout** *value*]] [**routing**] [**routing-type** *routing-number*] [**sequence** *value*] [**time-range** *name*] **Example:** `Device(config-ipv6-acl)# permit icmp any any hbh`	Sets permit conditions for the IPv6 ACL.
Step 5	**deny** *protocol* {*source-ipv6-prefix/prefix-length* | **any**| **host** *source-ipv6-address* | **auth**} [*operator*[*port-number*]] {*destination-ipv6-prefix/prefix-length* | **any** | **host** *destination-ipv6-address* | **auth**} [*operator* [*port-number*]] [**dest-option-type** [*doh-number* | *doh-type*]] [**dscp** *value*] [**flow-label** *value*] [**fragments**] [**hbh**] [**log**] [**log-input**] [**mobility**] [**mobility-type** [*mh-number* | *mh-type*]] [**routing**] [**routing-type** *routing-number*] [**sequence** *value*] [**time-range** *name*] [**undetermined-transport**] **Example:** `Device(config-ipv6-acl)# deny hbh any any`	Sets deny conditions for the IPv6 ACL.
Step 6	**end** **Example:** `Device (config-ipv6-acl)# end`	Returns to privileged EXEC configuration mode.

Configuration Example for IPv6 ACL Extensions for Hop by Hop Filtering

Example: IPv6 ACL Extensions for Hop by Hop Filtering

```
Device(config)# ipv6 access-list hbh_acl
Device(config-ipv6-acl)# permit tcp any any hbh
Device(config-ipv6-acl)# permit tcp any any
```

```
Device(config-ipv6-acl)# permit udp any any
Device(config-ipv6-acl)# permit udp any any hbh
Device(config-ipv6-acl)# permit hbh any any
Device(config-ipv6-acl)# permit any any
Device(config-ipv6-acl)# hardware statistics
Device(config-ipv6-acl)# exit

! Assign an IP address and add the ACL on the interface.

Device(config)# interface FastEthernet3/1
Device(config-if)# ipv6 address 1001::1/64
Device(config-if)# ipv6 traffic-filter hbh_acl in
Device(config-if)# exit
Device(config)# exit
Device# clear counters
Clear "show interface" counters on all interfaces [confirm]
Device#

! Verify the configurations.

Device# show running-config interface FastEthernet3/1

Building configuration...

Current configuration : 114 bytes
!
interface FastEthernet3/1
no switchport
ipv6 address 1001::1/64
ipv6 traffic-filter hbh_acl in counter
end
```

Additional References

Related Documents

Related Topic	Document Title
IPv6 addressing and connectivity	*IPv6 Configuration Guide*
Cisco IOS commands	Cisco IOS Master Commands List, All Releases
IPv6 commands	*IPv6 Command References*
Cisco IOS IPv6 features	*Cisco IOS IPv6 Feature Mapping*

Standards and RFCs

Standard/RFC	Title
RFCs for IPv6	*IPv6 RFCs*

Technical Assistance

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

Feature Information for IPv6 ACL Extensions for Hop by Hop Filtering

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Table 36 Feature Information for IPv6 ACL Extensions for Hop by Hop Filtering

Feature Name	Releases	Feature Information
IPv6 ACL Extensions for Hop by Hop Filtering	Cisco IOS Release XE 3.4S Cisco IOS Release XE 3.5S Cisco IOS Release XE 3.6S Cisco IOS Release XE 3.3SG	Allows you to control IPv6 traffic that might contain hop-by-hop extension headers. The following commands were introduced or modified: **deny** (IPv6), **permit** (IPv6).

Implementing Tunneling for IPv6

This module describes how to configure overlay tunneling techniques used by the Cisco IOS XE software to support the transition from IPv4-only networks to integrated IPv4- and IPv6-based networks. Tunneling encapsulates IPv6 packets in IPv4 packets and uses the IPv4 network as a link-layer mechanism.

Finding Feature Information

Your software release may not support all the features documented in this module. For the latest caveats and feature information, see Bug Search Tool and the release notes for your platform and software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the feature information table at the end of this module.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Restrictions for Implementing Tunneling for IPv6

- The IPv6 rapid deployment (6RD) feature is supported in an ethernet-only topology.
- IPv6 VRF is not supported with the 6RD feature.
- The Cisco ASR 1000 Series Aggregation Services Routers support as many as 2000 6RD tunnel interfaces.

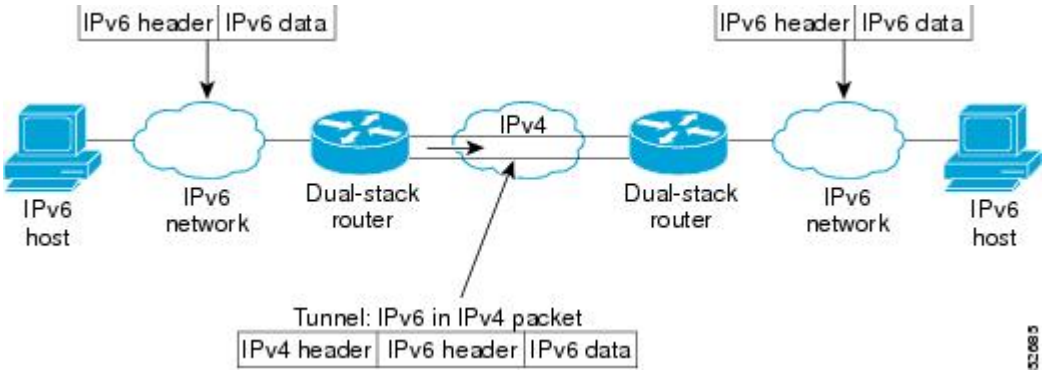
Information About Implementing Tunneling for IPv6

Overlay Tunnels for IPv6

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

- Manual
- Generic routing encapsulation (GRE)
- IPv4-compatible
- 6to4
- Intrasite Automatic Tunnel Addressing Protocol (ISATAP)

Figure 44 Overlay Tunnels



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

Use the table below to help you determine which type of tunnel you want to configure to carry IPv6 packets over an IPv4 network.

Table 37 Suggested Usage of Tunnel Types to Carry IPv6 Packets over an IPv4 Network

Tunneling Type	Suggested Usage	Usage Notes
Manual	Simple point-to-point tunnels that can be used within a site or between sites	Can carry IPv6 packets only.

Tunneling Type	Suggested Usage	Usage Notes
GRE- and IPv4- compatible	Simple point-to-point tunnels that can be used within a site or between sites	Can carry IPv6, Connectionless Network Service (CLNS), and many other types of packets.
IPv4- compatible	Point-to-multipoint tunnels	Uses the ::/96 prefix. We do not now recommend using this tunnel type.
6to4	Point-to-multipoint tunnels that can be used to connect isolated IPv6 sites	Sites use addresses from the 2002::/16 prefix.
6RD	IPv6 service is provided to customers over an IPv4 network by using encapsulation of IPv6 in IPv4.	Prefixes can be from the SP's own address block.
ISATAP	Point-to-multipoint tunnels that can be used to connect systems within a site	Sites can use any IPv6 unicast addresses.

Individual tunnel types are discussed in detail in this document. We recommend that you review and understand the information about the specific tunnel type that you want to implement. When you are familiar with the type of tunnel you need, see the table below for a summary of the tunnel configuration parameters that you may find useful.

Table 38 Tunnel Configuration Parameters by Tunneling Type

Tunneling Type	Tunnel Configuration Parameter			
Tunnel Mode	Tunnel Source	Tunnel Destination	Interface Prefix or Address	
Manual	ipv6ip	An IPv4 address, or a reference to an interface on which IPv4 is configured.	An IPv4 address.	An IPv6 address.
GRE/IPv4	gre ip		An IPv4 address.	An IPv6 address.
IPv4- compatible	ipv6ip auto-tunnel		Not required. These are all point-to-multipoint tunneling types. The IPv4 destination address is calculated, on a per-packet basis, from the IPv6 destination.	Not required. The interface address is generated as ::*tunnel-source*/96.
6to4	ipv6ip 6to4			An IPv6 address. The prefix must embed the tunnel source IPv4 address

Tunneling Type	Tunnel Configuration Parameter	
6RD	ipv6ip 6rd	An IPv6 address.
ISATAP	ipv6ip isatap	An IPv6 prefix in modified eui-64 format. The IPv6 address is generated from the prefix and the tunnel source IPv4 address.

IPv6 Manually Configured Tunnels

A manually configured tunnel is equivalent to a permanent link between two IPv6 domains over an IPv4 backbone. The primary use is for stable connections that require regular secure communication between two edge routers or between an end system and an edge router, or for connection to remote IPv6 networks.

An IPv6 address is manually configured on a tunnel interface, and manually configured IPv4 addresses are assigned to the tunnel source and the tunnel destination. The host or router at each end of a configured tunnel must support both the IPv4 and IPv6 protocol stacks. Manually configured tunnels can be configured between border routers or between a border router and a host.

GRE IPv4 Tunnel Support for IPv6 Traffic

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

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

GRE has a protocol field that identifies the passenger protocol. GRE tunnels allow Intermediate System-to-Intermediate System (IS-IS) or IPv6 to be specified as a passenger protocol, which allows both IS-IS and IPv6 traffic to run over the same tunnel. If GRE did not have a protocol field, it would be impossible to distinguish whether the tunnel was carrying IS-IS or IPv6 packets. The GRE protocol field is why it is desirable that you tunnel IS-IS and IPv6 inside GRE.

Automatic 6to4 Tunnels

An automatic 6to4 tunnel allows isolated IPv6 domains to be connected over an IPv4 network to remote IPv6 networks. The key difference between automatic 6to4 tunnels and manually configured tunnels is that the tunnel is not point-to-point; it is point-to-multipoint. In automatic 6to4 tunnels, routers are not configured in pairs because they treat the IPv4 infrastructure as a virtual nonbroadcast multiaccess (NBMA) link. The IPv4 address embedded in the IPv6 address is used to find the other end of the automatic tunnel.

An automatic 6to4 tunnel may be configured on a border router in an isolated IPv6 network, which creates a tunnel on a per-packet basis to a border router in another IPv6 network over an IPv4 infrastructure. The tunnel destination is determined by the IPv4 address of the border router extracted from the IPv6 address that starts with the prefix 2002::/16, where the format is 2002:*border-router-IPv4-address* ::/48. Following the embedded IPv4 address are 16 bits that can be used to number networks within the site. The border router at each end of a 6to4 tunnel must support both the IPv4 and IPv6 protocol stacks. 6to4 tunnels are configured between border routers or between a border router and a host.

The simplest deployment scenario for 6to4 tunnels is to interconnect multiple IPv6 sites, each of which has at least one connection to a shared IPv4 network. This IPv4 network could be the global Internet or a corporate backbone. The key requirement is that each site have a globally unique IPv4 address; the Cisco software uses this address to construct a globally unique 6to4/48 IPv6 prefix. As with other tunnel mechanisms, appropriate entries in a Domain Name System (DNS) that map between hostnames and IP addresses for both IPv4 and IPv6 allow the applications to choose the required address.

IPv6 Rapid Deployment Tunnels

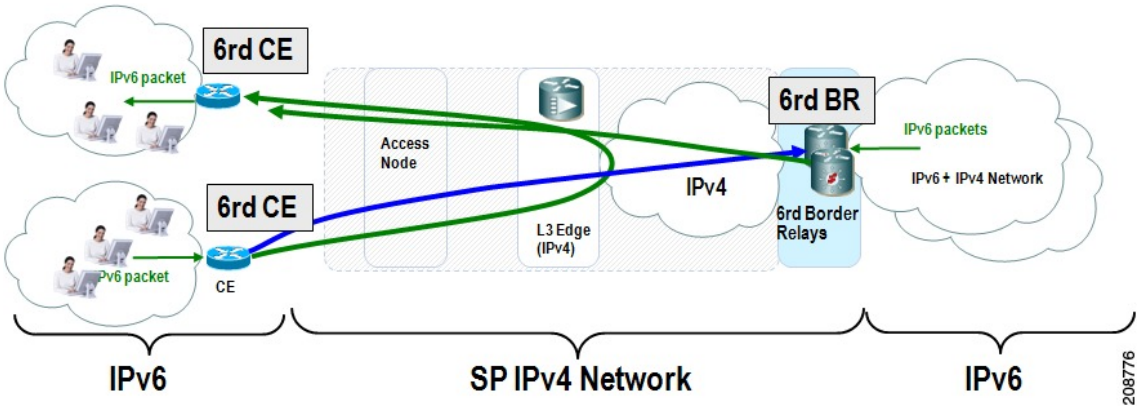
The 6RD feature is an extension of the 6to4 feature. The 6RD feature allows a service provider (SP) to provide a unicast IPv6 service to customers over its IPv4 network by using encapsulation of IPv6 in IPv4.

The main differences between 6RD and 6to4 tunneling are as follows:

- 6RD does not require addresses to have a 2002::/16 prefix; therefore, the prefix can be from the SP's own address block. This function allows the 6RD operational domain to be within the SP network. From the perspective of customer sites and the general IPv6 internet connected to a 6RD-enabled SP network, the IPv6 service provided is equivalent to native IPv6.
- All 32 bits of the IPv4 destination need not be carried in the IPv6 payload header. The IPv4 destination is obtained from a combination of bits in the payload header and information on the router. Furthermore, the IPv4 address is not at a fixed location in the IPv6 header as it is in 6to4.

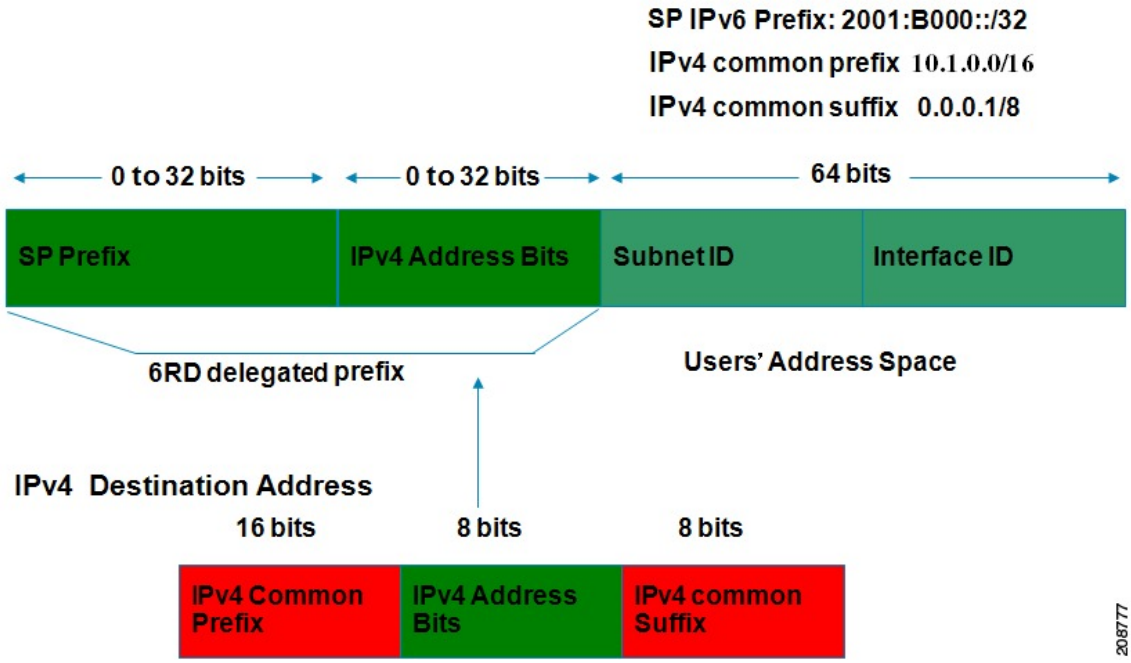
The 6RD SP prefix was selected by the SP for the IPv6 deployment shown in the figure below. The 6RD delegated prefix is derived from the SP prefix and the IPv4 address bits, and is used by the CE for hosts within its site.

Figure 45 6RD Deployment



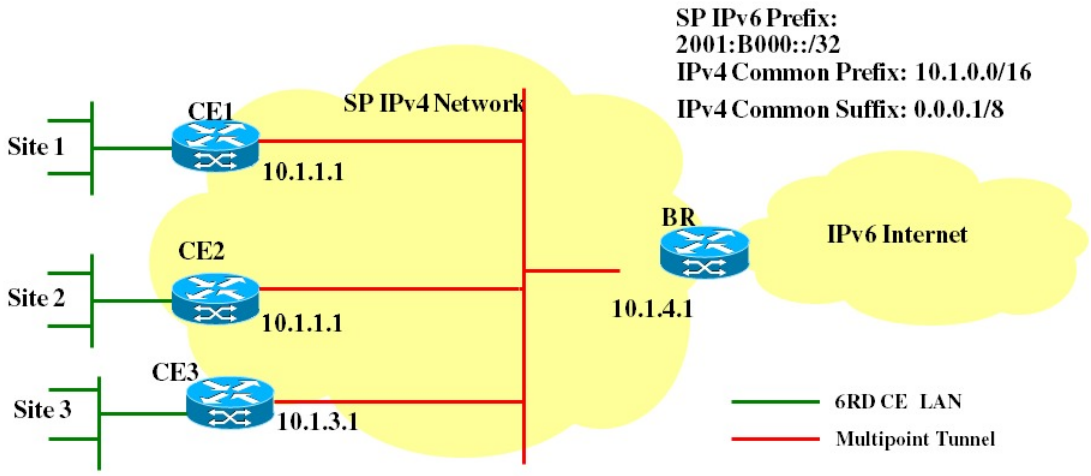
The figure below shows how 6RD prefix delegation works.

Figure 46 6RD Prefix Delegation Explanation



The figure below shows a 6RD prefix delegation topology.

Figure 47 6RD Prefix Delegation and Explanation



SP Prefix	2001:B000::/32
IPv4 Common Prefix	10.1.0.0/16
IPv4 Common Suffix	0.0.0.1/8
CE1: Delegated 6RD prefix	2001:B000:0100::/40
CE2: Delegated 6RD prefix	2001:B000:0200::/40
BR: Delegated 6RD prefix	2001:B000:0400::/40
CE1 (IPv4) tunnel transport source	10.1.1.1
CE2 (IPv4) tunnel transport source	10.1.2.1
BR (IPv4) tunnel transport source	10.1.4.1

208778

ISATAP Tunnels

ISATAP is an automatic overlay tunneling mechanism that uses the underlying IPv4 network as a NBMA link layer for IPv6. ISATAP is designed for transporting IPv6 packets *within* a site where a native IPv6 infrastructure is not yet available; for example, when sparse IPv6 hosts are deployed for testing. ISATAP tunnels allow individual IPv4 or IPv6 dual-stack hosts within a site to communicate with other such hosts on the same virtual link, basically creating an IPv6 network using the IPv4 infrastructure.

The ISATAP router provides standard router advertisement network configuration support for the ISATAP site. This feature allows clients to automatically configure themselves as they would do if they were connected to a GigabitEthernet or FastEthernet. It can also be configured to provide connectivity out of the site. ISATAP uses a well-defined IPv6 address format composed of any unicast IPv6 prefix (/64), which can be link local, or global (including 6to4 prefixes), enabling IPv6 routing locally or on the Internet. The IPv4 address is encoded in the last 32 bits of the IPv6 address, enabling automatic IPv6-in-IPv4 tunneling.

Although the ISATAP tunneling mechanism is similar to other automatic tunneling mechanisms, such as IPv6 6to4 tunneling, ISATAP is designed for transporting IPv6 packets *within* a site, not *between* sites.

ISATAP uses unicast addresses that include a 64-bit IPv6 prefix and a 64-bit interface identifier. The interface identifier is created in modified EUI-64 format in which the first 32 bits contain the value

000:5EFE to indicate that the address is an IPv6 ISATAP address. The table below describes an ISATAP address format.

Table 39 IPv6 ISATAP Address Format

64 Bits	32 Bits	32 Bits
link local or global IPv6 unicast prefix	0000:5EFE	IPv4 address of the ISATAP link

As shown in the table above, an ISATAP address consists of an IPv6 prefix and the ISATAP interface identifier. This interface identifier includes the IPv4 address of the underlying IPv4 link. The following example shows what an actual ISATAP address would look like if the prefix is 2001:DB8:1234:5678::/64 and the embedded IPv4 address is 10.173.129.8. In the ISATAP address, the IPv4 address is expressed in hexadecimal as 0AAD:8108:

2001:DB8:1234:5678:0000:5EFE:0AAD:8108

How to Implement Tunneling for IPv6

Configuring Manual IPv6 Tunnels

With manually configured IPv6 tunnels, an IPv6 address is configured on a tunnel interface, and manually configured IPv4 addresses are assigned to the tunnel source and the tunnel destination. The host or router at each end of a configured tunnel must support both the IPv4 and IPv6 protocol stacks.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface tunnel** *tunnel-number*
4. **ipv6 address** {*ipv6-address / prefix-length* | *prefix-name sub-bits/prefix-length*
5. **tunnel source** {*ip-address*| *interface-t ype interface-number*}
6. **tunnel destination** *ip-address*
7. **tunnel mode ipv6ip** [**6rd** | **6to4** | **auto-tunnel** | **isatap**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**interface tunnel** *tunnel-number* **Example:** `Router(config)# interface tunnel 0`	Specifies a tunnel interface and number, and enters interface configuration mode.
Step 4	**ipv6 address** {*ipv6-address* / *prefix-length* | *prefix-name sub-bits*/*prefix-length* **Example:** `Router(config-if)# ipv6 address 3ffe:b00:c18:1::3/127`	Specifies the IPv6 network assigned to the interface and enables IPv6 processing on the interface. **Note** Refer to the Implementing IPv6 Addressing and Basic Connectivity module for more information on configuring IPv6 addresses.
Step 5	**tunnel source** {*ip-address*| *interface-t ype interface-number*} **Example:** `Router(config-if)# tunnel source gigabitethernet 0/0/0`	Specifies the source IPv4 address or the source interface type and number for the tunnel interface. • If an interface is specified, the interface must be configured with an IPv4 address.
Step 6	**tunnel destination** *ip-address* **Example:** `Router(config-if)# tunnel destination 192.168.30.1`	Specifies the destination IPv4 address or hostname for the tunnel interface.

	Command or Action	Purpose
Step 7	**tunnel mode ipv6ip** [**6rd** | **6to4** | **auto-tunnel** | **isatap**	Specifies a manual IPv6 tunnel. **Note** The **tunnel mode ipv6ip** command specifies IPv6 as the passenger protocol and IPv4 as both the encapsulation and transport protocol for the manual IPv6 tunnel. • The **auto-tunnel** keyword is not supported on Cisco ASR 1000 series routers.
	Example: `Router(config-if)# tunnel mode ipv6ip`	

Configuring GRE IPv6 Tunnels

GRE tunnels can be configured to run over an IPv6 network layer and to transport IPv6 packets in IPv6 tunnels and IPv4 packets in IPv6 tunnels.

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

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface tunnel** *tunnel-number*
4. **ipv6 address** {*ipv6-address / prefix-length* | *prefix-name sub-bits*/*prefix-length*
5. **tunnel source** {*ip-address* | *ipv6-address* | *interface-type interface-number*}
6. **tunnel destination** {*host-name* | *ip-address* | *ipv6-address*}
7. **tunnel mode** {**aurp** | **cayman** | **dvmrp** | **eon** | **gre**| **gre multipoint** | **gre ipv6** | **ipip** [**decapsulate-any**] | **iptalk** | **ipv6** | **mpls** | **nos**

DETAILED STEPS

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

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

Configuring Automatic 6to4 Tunnels

With 6to4 tunnels, the tunnel destination is determined by the border router IPv4 address, which is concatenated to the prefix 2002::/16 in the format 2002:*border-router-IPv4-address* ::/48. The border router at each end of a 6to4 tunnel must support both the IPv4 and IPv6 protocol stacks.

Note The configuration of only one IPv4-compatible tunnel and one 6to4 IPv6 tunnel is supported on a router. If you choose to configure both of those tunnel types on the same router, we strongly recommend that they do not share the same tunnel source.

The reason that a 6to4 tunnel and an IPv4-compatible tunnel cannot share an interface is that both of them are NBMA "point-to-multipoint" access links and only the tunnel source can be used to reorder the packets from a multiplexed packet stream into a single packet stream for an incoming interface. So when a packet with an IPv4 protocol type of 41 arrives on an interface, that packet is mapped to an IPv6 tunnel interface based on the IPv4 address. However, if both the 6to4 tunnel and the IPv4-compatible tunnel share the same source interface, the router is not able to determine the IPv6 tunnel interface to which it should assign the incoming packet.

IPv6 manually configured tunnels can share the same source interface because a manual tunnel is a "point-to-point" link, and both the IPv4 source and IPv4 destination of the tunnel are defined.

>

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface tunnel** *tunnel-number*
4. **ipv6 address** {*ipv6-address / prefix-length* | *prefix-name sub-bits/prefix-length*
5. **tunnel source** {*ip-address*| *interface-t ype interface-number*}
6. **tunnel mode ipv6ip** [**6rd** | **6to4** | **auto-tunnel** | **isatap**
7. **exit**
8. **ipv6 route** [**vrf** *vrf-name*] *ipv6-prefix / prefix-length*{*ipv6-address* | *interface-type interface-number* [*ipv6-address*]} [**nexthop-vrf** [*vrf-name1* | **default**]] [*administrative-distance*] [*administrative-multicast-distance* | **unicast** | **multicast**] [*next-hop-address*] [**tag** *tag*]

DETAILED STEPS

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

	Command or Action	Purpose
Step 3	**interface tunnel** *tunnel-number* **Example:** `Router(config)# interface tunnel 1`	Specifies a tunnel interface and number, and enters interface configuration mode.
Step 4	**ipv6 address** {*ipv6-address* / *prefix-length* | *prefix-name sub-bits*/*prefix-length* **Example:** `Router(config-if)# ipv6 address 3ffe:b00:c18:1::3/127`	Specifies the IPv6 network assigned to the interface and enables IPv6 processing on the interface.
Step 5	**tunnel source** {*ip-address*| *interface-t ype interface-number*} **Example:** `Router(config-if)# tunnel source loopback 1`	Specifies the source interface type and number for the tunnel interface.
Step 6	**tunnel mode ipv6ip** [**6rd** | **6to4** | **auto-tunnel** | **isatap** **Example:** `Router(config-if)# tunnel mode ipv6ip 6rd`	Configures a static IPv6 tunnel interface. • The **auto-tunnel** keyword is not supported on Cisco ASR 1000 series routers.
Step 7	**exit** **Example:** `Router(config-if) exit`	Exits interface configuration mode, and enters global configuration mode.
Step 8	**ipv6 route** [**vrf** *vrf-name*] *ipv6-prefix* / *prefix-length*{*ipv6-address* | *interface-type interface-number* [*ipv6-address*]} [**nexthop-vrf** [*vrf-name1* | **default**]] [*administrative-distance*] [*administrative-multicast-distance* | **unicast** | **multicast**] [*next-hop-address*] [**tag** *tag*] **Example:** `Router(config)# ipv6 route 2002::/16 tunnel 0`	Configures a static route for the IPv6 6to4 prefix 2002::/16 to the specified tunnel interface. **Note** When configuring a 6to4 overlay tunnel, you must configure a static route for the IPv6 6to4 prefix 2002::/16 to the 6to4 tunnel interface. • The tunnel number specified in the **ipv6 route** command must be the same tunnel number specified in the **interface tunnel** command.

Configuring 6RD Tunnels

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface tunnel** *tunnel-number*
4. **tunnel source** {*ip-address*| *interface-t ype interface-number*}
5. **tunnel mode ipv6ip** [**6rd** | **6to4** | **auto-tunnel** | **isatap**]
6. **tunnel 6rd prefix** *ipv6-prefix* / *prefix-length*
7. **tunnel 6rd ipv4** {**prefix-length** *length*} {**suffix-length** *length*}

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**interface tunnel** *tunnel-number* **Example:** `Router(config)# interface tunnel 1`	Specifies a tunnel interface and number, and enters interface configuration mode.
Step 4	**tunnel source** {*ip-address*| *interface-t ype interface-number*} **Example:** `Router(config-if)# tunnel source loopback 1`	Specifies the source interface type and number for the tunnel interface.
Step 5	**tunnel mode ipv6ip** [**6rd**	**6to4**	**auto-tunnel**	**isatap**] **Example:** `Router(config-if)# tunnel mode ipv6ip 6rd`	Configures a static IPv6 tunnel interface. • The **auto-tunnel** keyword is not supported on Cisco ASR 1000 series routers.

	Command or Action	Purpose
Step 6	**tunnel 6rd prefix** *ipv6-prefix* / *prefix-length* **Example:** `Router(config-if)# tunnel 6rd prefix 2001:B000::/32`	Specifies the common IPv6 prefix on IPv6 rapid 6RD tunnels.
Step 7	**tunnel 6rd ipv4** {**prefix-length** *length*} {**suffix-length** *length*} **Example:** `Router(config-if)# tunnel 6rd ipv4 prefix-length 16 suffix 8`	Specifies the prefix length and suffix length of the IPv4 transport address common to all the 6RD routers in a domain.

Configuring ISATAP Tunnels

The **tunnel source** command used in the configuration of an ISATAP tunnel must point to an interface with an IPv4 address configured. The ISATAP IPv6 address and prefix (or prefixes) advertised are configured as for a native IPv6 interface. The IPv6 tunnel interface must be configured with a modified EUI-64 address because the last 32 bits in the interface identifier are constructed using the IPv4 tunnel source address.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface tunnel** *tunnel-number*
4. **ipv6 address** {*ipv6-address* / *prefix-length* | *prefix-name sub-bits*/*prefix-length*
5. **no ipv6 nd ra suppress**
6. **tunnel source** {*ip-address*| *interface-type interface-number*}
7. **tunnel mode ipv6ip** [**6rd** | **6to4** | **auto-tunnel** | **isatap**

DETAILED STEPS

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

Command or Action	Purpose
Step 2 **configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3 **interface tunnel** *tunnel-number* **Example:** `Router(config)# interface tunnel 1`	Specifies a tunnel interface and number, and enters interface configuration mode.
Step 4 **ipv6 address** {*ipv6-address* / *prefix-length* | *prefix-name sub-bits*/*prefix-length* **Example:** `Router(config-if)# ipv6 address 2001:DB8:6301::/64 eui-64`	Specifies the IPv6 address assigned to the interface and enables IPv6 processing on the interface.
Step 5 **no ipv6 nd ra suppress** **Example:** `Router(config-if)# no ipv6 nd ra suppress`	Sending of IPv6 router advertisements is disabled by default on tunnel interfaces. This command reenables the sending of IPv6 router advertisements to allow client autoconfiguration.
Step 6 **tunnel source** {*ip-address*| *interface-type interface-number*} **Example:** `Router(config-if)# tunnel source gigabitethernet 1/0/1`	Specifies the source interface type and number for the tunnel interface. **Note** The interface type and number specified in the **tunnel source** command must be configured with an IPv4 address.
Step 7 **tunnel mode ipv6ip** [**6rd** | **6to4** | **auto-tunnel** | **isatap** **Example:** `Router(config-if)# tunnel mode ipv6ip isatap`	Specifies an IPv6 overlay tunnel using a ISATAP address. • The **auto-tunnel** keyword is not supported on Cisco ASR 1000 series routers.

Verifying IPv6 Tunnel Configuration and Operation

SUMMARY STEPS

1. **enable**
2. **show interfaces tunnel** *number* [**accounting**]
3. **ping** [*protocol*] *destination*
4. **show ip route** [*address*[*mask*]]
5. **show tunnel 6rd** [*interface-type interface-number*
6. **show tunnel 6rd destination** *ipv6-prefix tunnel-interface interface-number*
7. **show tunnel 6rd prefix** *ipv4-destination tunnel-interface interface-number*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**show interfaces tunnel** *number* [**accounting**] **Example:** `Router# show interfaces tunnel 0`	(Optional) Displays tunnel interface information. • Use the *number* argument to display information for a specified tunnel.
Step 3	**ping** [*protocol*] *destination* **Example:** `Router# ping 10.0.0.1`	(Optional) Diagnoses basic network connectivity.
Step 4	**show ip route** [*address*[*mask*]] **Example:** `Router# show ip route 10.0.0.2`	(Optional) Displays the current state of the routing table. **Note** Only the syntax relevant for this task is shown.
Step 5	**show tunnel 6rd** [*interface-type interface-number* **Example:** `Router# show tunnel 6rd tunnel 1`	Displays 6RD information about a tunnel.

Command or Action	Purpose
Step 6 **show tunnel 6rd destination** *ipv6-prefix tunnel-interface interface-number* **Example:** `Router# show tunnel 6rd destination 2001:B000:300:: tunnel 1`	Translates and displays a 6RD prefix to the corresponding IPv4 destination.
Step 7 **show tunnel 6rd prefix** *ipv4-destination tunnel-interface interface-number* **Example:** `Router# show tunnel 6rd prefix 10.1.4.1 tunnel 1`	Translates and displays an IPv4 destination address to the corresponding 6RD prefix.

- Examples, page 644

Examples

- Sample Output from the show interfaces tunnel Command, page 644
- Sample Output from the ping Command When Checking the Local Endpoint, page 645
- Sample Output from the show ip route Command, page 645
- Sample Output from the ping Command When Checking the Remote Endpoint, page 645

Sample Output from the show interfaces tunnel Command

This example uses a generic example suitable for both IPv6 manually configured tunnels and IPv6 over IPv4 GRE tunnels. In the example, two routers are configured to be endpoints of a tunnel. Router A has GigabitEthernet interface 0/0/0 configured as tunnel interface 0 with an IPv4 address of 10.0.0.1 and an IPv6 prefix of 2001:DB8:1111:2222::1/64. Router B has GigabitEthernet interface 0/0/0 configured as tunnel interface 1 with an IPv4 address of 10.0.0.2 and an IPv6 prefix of 2001:DB8:1111:2222::2/64. To verify that the tunnel source and destination addresses are configured, use the **show interfaces tunnel** command on Router A.

```
RouterA# show interfaces tunnel 0
Tunnel0 is up, line protocol is up
  Hardware is Tunnel
  MTU 1514 bytes, BW 9 Kbit, DLY 500000 usec,
     reliability 255/255, txload 1/255, rxload 1/255
  Encapsulation TUNNEL, loopback not set
  Keepalive not set
  Tunnel source 10.0.0.1 (GigabitEthernet0/0/0), destination 10.0.0.2, fastswitch TTL 255
  Tunnel protocol/transport GRE/IP, key disabled, sequencing disabled
  Tunnel TTL 255
  Checksumming of packets disabled,  fast tunneling enabled
  Last input 00:00:14, output 00:00:04, output hang never
  Last clearing of "show interface" counters never
  Input queue: 0/75/0/0 (size/max/drops/flushes); Total output drops: 0
  Queueing strategy: fifo
  Output queue :0/0 (size/max)
  5 minute input rate 0 bits/sec, 0 packets/sec
  5 minute output rate 0 bits/sec, 0 packets/sec
```

```
     4 packets input, 352 bytes, 0 no buffer
     Received 0 broadcasts, 0 runts, 0 giants, 0 throttles
     0 input errors, 0 CRC, 0 frame, 0 overrun, 0 ignored, 0 abort
     8 packets output, 704 bytes, 0 underruns
     0 output errors, 0 collisions, 0 interface resets
     0 output buffer failures, 0 output buffers swapped out
```

Sample Output from the ping Command When Checking the Local Endpoint

To check that the local endpoint is configured and working, use the **ping** command on Router A:

```
RouterA# ping 2001:DB8:1111:2222::2
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 2001:DB8:1111:2222::2, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 20/20/20 ms
```

Sample Output from the show ip route Command

To check that a route exists to the remote endpoint address, use the **show ip route** command:

```
RouterA# show ip route 10.0.0.2
Routing entry for 10.0.0.0/24
  Known via "connected", distance 0, metric 0 (connected, via interface)
  Routing Descriptor Blocks:
  * directly connected, via GigabitEthernet0/0/0
      Route metric is 0, traffic share count is 1
```

Sample Output from the ping Command When Checking the Remote Endpoint

To check that the remote endpoint address is reachable, use the **ping** command on Router A.

Note The remote endpoint address may not be reachable using the **ping** command because of filtering, but the tunnel traffic may still reach its destination.

```
RouterA# ping 10.0.0.2
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 10.0.0.2, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 20/21/28 ms
```

To check that the remote IPv6 tunnel endpoint is reachable, use the **ping** command again on Router A. The same note on filtering also applies to this example.

```
RouterA# ping 1::2
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 1::2, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 20/20/20 ms
```

These steps may be repeated at the other endpoint of the tunnel.

Configuration Examples for Implementing Tunneling for IPv6

- Example: Configuring Manual IPv6 Tunnels, page 646
- Example Configuring GRE Tunnels, page 646
- Example: Configuring 6to4 Tunnels, page 648

Example: Configuring Manual IPv6 Tunnels

The following example configures a manual IPv6 tunnel between router A and router B. In the example, tunnel interface 0 for both router A and router B is manually configured with a global IPv6 address. The tunnel source and destination addresses are also manually configured.

Router A Configuration

```
interface gigabitethernet 0/0/0
 ip address 192.168.99.1 255.255.255.0
interface tunnel 0
 ipv6 address 3ffe:b00:c18:1::3/127
 tunnel source gigabitethernet 0/0/0
 tunnel destination 2001:DB8:1111:2222::1/64
 tunnel mode ipv6ip
```

Router B Configuration

```
interface gigabitethernet 0/0/0
 ip address 192.168.30.1 255.255.255.0
interface tunnel 0
 ipv6 address 3ffe:b00:c18:1::2/127
 tunnel source gigabitethernet 0/0/0
 tunnel destination 2001:DB8:1111:2222::2/64
 tunnel mode ipv6ip
```

Example Configuring GRE Tunnels

The following example configures a GRE tunnel running both IS-IS and IPv6 traffic between router A and router B:

Router A Configuration

```
ipv6 unicast-routing
clns routing
!
interface tunnel 0
 no ip address
 ipv6 address 2001:DB8:1111:2222::1/64
 ipv6 router isis
 tunnel source GigabitEthernet 0/0/0
 tunnel destination 10.0.0.2
 tunnel mode gre ipv6
!
interface GigabitEthernet0/0/0
 ip address 10.0.0.1 255.255.255.0
!
router isis
 net 49.0000.0000.000a.00
```

Router B Configuration

```
ipv6 unicast-routing
clns routing
!
interface tunnel 0
```

```
 no ip address
 ipv6 address 2001:DB8:1111:2222::2/64
 ipv6 router isis
 tunnel source GigabitEthernet 0/0/0
 tunnel destination 10.0.0.1
 tunnel mode gre ipv6
!
interface GigabitEthernet0/0/0
 ip address 10.0.0.2 255.255.255.0
!
router isis
 net 49.0000.0000.000b.00
 address-family ipv6
 redistribute static
 exit-address-family
```

- Example: Tunnel Destination Address for IPv6 Tunnel, page 647

Example: Tunnel Destination Address for IPv6 Tunnel

```
Router(config
)
#
interface Tunnel0
Router(config
-if)
#
ipv6 address 2001:1:1::1/48
Router(config
-if)
#
tunnel source GigabitEthernet 0/0/0
Router(config
-if)
#
tunnel destination 10.0.0.2
Router(config
-if)
#
tunnel mode gre ipv6
Router(config
-if)
#
exit
!
Router(config
)
#
interface GigabitEthernet0/0/0
Router(config
-if)
#
ip address 10.0.0.1 255.255.255.0
Router(config
-if)
#
exit
!
Router(config
)
#
ipv6 unicast-routing
Router(config
)
#
router isis

Router(config
)
#
net 49.0000.0000.000a.00
```

Example: Configuring 6to4 Tunnels

The following example configures a 6to4 tunnel on a border router in an isolated IPv6 network. The IPv4 address is 192.168.99.1, which translates to the IPv6 prefix of 2002:c0a8:6301::/48. The IPv6 prefix is subnetted into 2002:c0a8:6301::/64 for the tunnel interface: 2002:c0a8:6301:1::/64 for the first IPv6 network, and 2002:c0a8:6301:2::/64 for the second IPv6 network. The static route ensures that any other traffic for the IPv6 prefix 2002::/16 is directed to tunnel interface 0 for automatic tunneling.

```
interface GigabitEthernet0/0/0
 description IPv4 uplink
 ip address 192.168.99.1 255.255.255.0
!
interface GigabitEthernet1/0/0
 description IPv6 local network 1
 ipv6 address 2002:c0a8:6301:1::1/64
!
interface GigabitEthernet2/0/0
 description IPv6 local network 2
 ipv6 address 2002:c0a8:6301:2::1/64
!
interface Tunnel0
 description IPv6 uplink
 no ip address
 ipv6 address 2002:c0a8:6301::1/64
 tunnel source GigabitEthernet0/0/0
 tunnel mode ipv6ip 6to4
!
ipv6 route 2002::/16 tunnel 0
```

Example: Configuring 6RD Tunnels

The following example shows the running configuration of a 6RD tunnel and the corresponding output of the **show tunnel 6rd** command:

```
interface Tunnel1
 ipv6 address 2001:B000:100::1/32
 tunnel source loopback 1
 tunnel mode ipv6ip 6rd
 tunnel 6rd prefix 2001:B000::/32
 tunnel 6rd ipv4 prefix-len 16 suffix-len 8
end
Router# show tunnel 6rd tunnel 1
Interface Tunnel1:
  Tunnel Source: 10.1.1.1
  6RD: Operational, V6 Prefix: 2001:B000::/32
  V4 Common Prefix Length: 16, Value: 10.1.0.0
  V4 Common Suffix Length: 8, Value: 0.0.0.1
```

Example: Configuring IPv4-Compatible IPv6 Tunnels

The following example configures an IPv4-compatible IPv6 tunnel that allows Border Gateway Protocol (BGP) to run between a number of routers without having to configure a mesh of manual tunnels. Each router has a single IPv4-compatible tunnel, and multiple BGP sessions can run over each tunnel, one to each neighbor. GigabitEthernet interface 0/0/0 is used as the tunnel source. The tunnel destination is automatically determined by the IPv4 address in the low-order 32 bits of an IPv4-compatible IPv6 address. Specifically, the IPv6 prefix 0:0:0:0:0:0 is concatenated to an IPv4 address (in the format 0:0:0:0:0:A.B.C.D or ::A.B.C.D) to create the IPv4-compatible IPv6 address. GigabitEthernet interface 0/0/0 is configured with a global IPv6 address and an IPv4 address (the interface supports both the IPv6 and IPv4 protocol stacks).

Multiprotocol BGP is used in the example to exchange IPv6 reachability information with the peer 10.67.0.2. The IPv4 address of GigabitEthernet interface 0/0/0 is used in the low-order 32 bits of an IPv4-

compatible IPv6 address and is also used as the next-hop attribute. Using an IPv4-compatible IPv6 address for the BGP neighbor allows the IPv6 BGP session to be automatically transported over an IPv4-compatible tunnel.

```
interface tunnel 0
 tunnel source GigabitEthernet 0/0/0
 tunnel mode ipv6ip auto-tunnel
interface Gigabitethernet 0/0/0
 ip address 10.27.0.1 255.255.255.0
 ipv6 address 3000:2222::1/64
router bgp 65000
 no synchronization
 no bgp default ipv4-unicast
neighbor ::10.67.0.2 remote-as 65002
address-family ipv6
 neighbor ::10.67.0.2 activate
 neighbor ::10.67.0.2 next-hop-self
 network 2001:2222:d00d:b10b::/64
```

Example: Configuring ISATAP Tunnels

The following example shows the tunnel source defined on GigabitEthernet 0/0/0 and the **tunnel mode** command used to configure the ISATAP tunnel. Router advertisements are enabled to allow client autoconfiguration.

```
ipv6 unicast-routing
interface tunnel 1
 tunnel source Gigabitethernet 0/0/0
 tunnel mode ipv6ip isatap
 ipv6 address 2001:DB8::/64 eui-64
 no ipv6 nd ra suppress
 exit
```

Additional References

Related Documents

Related Topic	Document Title
IPv6 addressing and connectivity	*IPv6 Configuration Guide*
Cisco IOS commands	Master Commands List, All Releases
IPv6 commands	IPv6 Command Reference
Cisco IOS IPv6 features	IPv6 Feature Mapping

Standards and RFCs

Standard/RFC	Title
RFCs for IPv6	IPv6 RFCs

Technical Assistance

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

Feature Information for Implementing Tunneling for IPv6

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Table 40 Feature Information for Implementing Tunneling for IPv6

Feature Name	Releases	Feature Information
IPv6 Tunneling--6RD IPv6 Rapid Deployment	Cisco IOS XE Release 3.1S	The 6RD feature allows a service provider to provide a unicast IPv6 service to customers over its IPv4 network by using encapsulation of IPv6 in IPv4.
IPv6 Tunneling--Automatic 6to4 Tunnels	Cisco IOS XE Release 2.1	An automatic 6to4 tunnel allows isolated IPv6 domains to be connected over an IPv4 network to remote IPv6 networks.
IPv6 Tunneling--Automatic IPv4-Compatible Tunnels	Cisco IOS XE Release 2.1	Automatic IPv4-compatible tunnels use IPv4-compatible IPv6 addresses.
IPv6 Tunneling--IP over IPv6 GRE Tunnels	Cisco IOS XE Release 2.4	GRE tunnels are links between two points, with a separate tunnel for each link.
IPv6 Tunneling--IPv4 over IPv6 Tunnels	Cisco IOS XE Release 2.1	IPv6 supports this feature

Feature Name	Releases	Feature Information
IPv6 Tunneling--IPv6 over IPv4 GRE Tunnels	Cisco IOS XE Release 2.1	GRE tunnels are links between two points, with a separate tunnel for each link. The tunnels are not tied to a specific passenger or transport protocol, but in this case carry IPv6 as the passenger protocol with the GRE as the carrier protocol and IPv4 or IPv6 as the transport protocol.
IPv6 Tunneling--ISATAP Tunnel Support	Cisco IOS XE Release 2.1 Cisco IOS XE Release 3.3SG	ISATAP is an automatic overlay tunneling mechanism that uses the underlying IPv4 network as a NBMA link layer for IPv6.
IPv6 Tunneling--Manually Configured IPv6 over IPv4 Tunnels	Cisco IOS XE Release 2.1	A manually configured tunnel is equivalent to a permanent link between two IPv6 domains over an IPv4 backbone.
IPv6 Switching: CEFv6 Switched Configured IPv6 over IPv4 Tunnels	Cisco IOS XE Release 3.3SG	Supports CEF switching of IPv6 auto 6to4 tunnels.

IPv6 Virtual Fragmentation Reassembly

Finding Feature Information

Your software release may not support all the features documented in this module. For the latest caveats and feature information, see Bug Search Tool and the release notes for your platform and software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the feature information table at the end of this module.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Information About IPv6 Virtual Fragmentation Reassembly

IPv6 Virtual Fragmentation Reassembly

Fragmentation is a process of breaking down an IP datagram into smaller packets to be transmitted over different types of network media. Non-initial fragments of a fragmented IPv6 packet is used to pass through IPsec and NAT64 without any examination due to the lack of the L4 header, which usually is only available on the initial fragment. The IPv6 Virtual Fragmentation Reassembly (VFR) feature provides the ability to collect the fragments and provide L4 info for all fragments for IPsec and NAT64 features.

How to Implement IPv6 Virtual Fragmentation Reassembly

Configuring IPv6 Virtual Fragmentation Reassembly

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type number*
4. ipv6 virtual-reassembly [**in** | **out**] [**max-reassemblies** *maxreassemblies*] [**max-fragments** *max-fragments*] [**timeout** *seconds*] [**drop-fragments**
5. **exit**
6. show ipv6 virtual-reassembly interface *interface-type*
7. show ipv6 virtual-reassembly features interface *interface-type*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**interface** *type number* **Example:** `Router(config)# interface gigabitethernet 3/1/1`	Specifies an interface type and number, and places the router in interface configuration mode.
Step 4	ipv6 virtual-reassembly [**in** | **out**] [**max-reassemblies** *maxreassemblies*] [**max-fragments** *max-fragments*] [**timeout** *seconds*] [**drop-fragments** **Example:** `Router(config-if)# ipv6 virtual-reassembly max-reassemblies 32 max-fragments 4 timeout 7`	Enables VFR on an interface.

	Command or Action	Purpose
Step 5	**exit** **Example:** `Router(config-if)# exit`	Exits interface configuration mode and places the router in global configuration mode. • Enter this command twice to reach privileged EXEC mode.
Step 6	show ipv6 virtual-reassembly interface *interface-type* **Example:** `Router# show ipv6 virtual-reassembly interface e1/1/1`	Displays VRF configuration and statistical information on a specific interface.
Step 7	show ipv6 virtual-reassembly features interface *interface-type* **Example:** `Router# show ipv6 virtual-reassembly features`	Displays VFR information on all interfaces or on a specified interface.

Configuration Example for IPv6 Virtual Fragmentation Reassembly

Example: Configuring IPv6 Virtual Fragmentation Reassembly

```
Router# show ipv6 virtual-reassembly interface gigabitethernet1/1/1
GigabitEthernet1/1/1:
IPv6 Virtual Fragment Reassembly (VFR) is ENABLED(in)
Concurrent reassemblies (max-reassemblies): 64
Fragments per reassembly (max-fragments): 16
Reassembly timeout (timeout): 3 seconds
Drop fragments: OFF
Current reassembly count: 0
Current fragment count: 0
Total reassembly count: 6950
Total reassembly timeout count: 9
GigabitEthernet1/1/1:
IPv6 Virtual Fragment Reassembly (VFR) is ENABLED(out)
Concurrent reassemblies (max-reassemblies): 64
Fragments per reassembly (max-fragments): 16
Reassembly timeout (timeout): 3 seconds
Drop fragments: OFF
Current reassembly count: 0
Current fragment count: 0
Total reassembly count: 0
Total reassembly timeout count: 0
```

Additional References

Related Documents

Related Topic	Document Title
IPv6 addressing and connectivity	*IPv6 Configuration Guide*
Cisco IOS commands	Master Commands List, All Releases
IPv6 commands	IPv6 Command Reference
Cisco IOS IPv6 features	IPv6 Feature Mapping

Standards and RFCs

Standard/RFC	Title
RFCs for IPv6	IPv6 RFCs

Technical Assistance

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

Feature Information for IPv6 Virtual Fragmentation Reassembly

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Table 41 Feature Information for IPv6 Virtual Fragmentation Reassembly

Feature Name	Releases	Feature Information
IPv6 Virtual Fragmentation Reassembly	Cisco IOS XE Release 3.4S	The IPv6 VFR feature provides the ability to collect the fragments and provide L4 info for all fragments for IPsec and NAT64 features.

IPv6 RFCs

Standards and RFCs

RFCs	Title
RFC 1195	*Use of OSI IS-IS for Routing in TCP/IP and Dual Environments*
RFC 1267	*A Border Gateway Protocol 3 (BGP-3)*
RFC 1305	*Network Time Protocol (Version 3) Specification, Implementation and Analysis*
RFC 1583	*OSPF version 2*
RFC 1772	*Application of the Border Gateway Protocol in the Internet*
RFC 1886	*DNS Extensions to Support IP version 6*
RFC 1918	*Address Allocation for Private Internets*
RFC 1981	*Path MTU Discovery for IP version 6*
RFC 2080	*RIPng for IPv6*
RFC 2281	*Cisco Hot Standby Router Protocol (HSRP)*
RFC 2332	*NBMA Next Hop Resolution Protocol (NHRP)*
RFC 2373	*IP Version 6 Addressing Architecture*
RFC 2374	*An Aggregatable Global Unicast Address Format*
RFC 2375	*IPv6 Multicast Address Assignments*
RFC 2401	*Security Architecture for the Internet Protocol*
RFC 2402	*IP Authentication Header*
RFC 2404	*The Use of Hash Message Authentication Code Federal Information Processing Standard 180-1 within Encapsulating Security Payload and Authentication Header*

RFCs	Title
RFC 2406	*IP Encapsulating Security Payload (ESP)*
RFC 2407	*The Internet Security Domain of Interpretation for ISAKMP*
RFC 2408	*Internet Security Association and Key Management Protocol*
RFC 2409	*Internet Key Exchange (IKE)*
RFC 2427	*Multiprotocol Interconnect over Frame Relay*
RFC 2428	*FTP Extensions for IPv6 and NATs*
RFC 2460	*Internet Protocol, Version 6 (IPv6) Specification*
RFC 2461	*Neighbor Discovery for IP Version 6 (IPv6)*
RFC 2462	*IPv6 Stateless Address Autoconfiguration*
RFC 2463	*Internet Control Message Protocol (ICMPv6) for the Internet Protocol Version 6 (IPv6) Specification*
RFC 2464	*Transmission of IPv6 Packets over Ethernet*
RFC 2467	*Transmission of IPv6 Packets over FDDI*
RFC 2472	*IP Version 6 over PPP*
RFC 2473	*Generic Packet Tunneling in IPv6 Specification*
RFC 2474	*Definition of the Differentiated Services Field (DS Field) in the IPv4 and IPv6 Headers*
RFC 2475	*An Architecture for Differentiated Services Framework*
RFC 2492	*IPv6 over ATM*
RFC 2545	*Use of BGP-4 Multiprotocol Extensions for IPv6 Inter-Domain Routing*
RFC 2590	*Transmission of IPv6 Packets over Frame Relay Specification*
RFC 2597	*Assured Forwarding PHB*
RFC 2598	*An Expedited Forwarding PHB*
RFC 2640	*Internet Protocol, Version 6 Specification*
RFC 2684	*Multiprotocol Encapsulation over ATM Adaptation Layer 5*

RFCs	Title
RFC 2697	*A Single Rate Three Color Marker*
RFC 2698	*A Two Rate Three Color Marker*
RFC 2710	*Multicast Listener Discovery (MLD) for IPv6*
RFC 2711	*IPv6 Router Alert Option*
RFC 2732	*Format for Literal IPv6 Addresses in URLs*
RFC 2765	*Stateless IP/ICMP Translation Algorithm (SIIT)*
RFC 2766	*Network Address Translation-Protocol Translation (NAT-PT)*
RFC 2858	*Multiprotocol Extensions for BGP-4*
RFC 2893	*Transition Mechanisms for IPv6 Hosts and Routers*
RFC 3056	*Connection of IPv6 Domains via IPv4 Clouds*
RFC 3068	*An Anycast Prefix for 6to4 Relay Routers*
RFC 3095	*RObust Header Compression (ROHC): Framework and Four Profiles: RTP, UDP, ESP, and Uncompressed*
RFC 3107	*Carrying Label Information in BGP-4*
RFC 3137	*OSPF Stub Router Advertisement*
RFC 3147	*Generic Routing Encapsulation over CLNS*
RFC 3152	*Delegation of IP6.ARPA*
RFC 3162	*RADIUS and IPv6*
RFC 3315	*Dynamic Host Configuration Protocol for IPv6 (DHCPv6)*
RFC 3319	*Dynamic Host Configuration Protocol (DHCPv6) Options for Session Initiated Protocol (SIP) Servers*
RFC 3392	*Capabilities Advertisement with BGP-4*
RFC 3414	*User-based Security Model (USM) for version 3 of the Simple Network Management Protocol (SNMPv3)*
RFC 3484	*Default Address Selection for Internet Protocol version 6 (IPv6)*

RFCs	Title
RFC 3513	*Internet Protocol Version 6 (IPv6) Addressing Architecture*
RFC 3576	*Change of Authorization*
RFC 3587	*IPv6 Global Unicast Address Format*
RFC 3590	*Source Address Selection for the Multicast Listener Discovery (MLD) Protocol*
RFC 3596	*DNS Extensions to Support IP Version 6*
RFC 3633	*DHCP IPv6 Prefix Delegation*
RFC 3646	*DNS Configuration options for Dynamic Host Configuration Protocol for IPv6 (DHCPv6)*
RFC 3697	*IPv6 Flow Label Specification*
RFC 3736	*Stateless DHCP Service for IPv6*
RFC 3756	*IPv6 Neighbor Discovery (ND) Trust Models and Threats*
RFC 3759	*RObust Header Compression (ROHC): Terminology and Channel Mapping Examples*
RFC 3775	*Mobility Support in IPv6*
RFC 3810	*Multicast Listener Discovery Version 2 (MLDv2) for IPv6*
RFC 3846	*Mobile IPv4 Extension for Carrying Network Access Identifiers*
RFC 3879	*Deprecating Site Local Addresses*
RFC 3898	*Network Information Service (NIS) Configuration Options for Dynamic Host Configuration Protocol for IPv6 (DHCPv6)*
RFC 3954	*Cisco Systems NetFlow Services Export Version 9*
RFC 3956	*Embedding the Rendezvous Point (RP) Address in an IPv6 Multicast Address*
RFC 3963	*Network Mobility (NEMO) Basic Support Protocol*
RFC 3971	*SEcure Neighbor Discovery (SEND)*
RFC 3972	*Cryptographically Generated Addresses (CGA)*
RFC 4007	*IPv6 Scoped Address Architecture*

RFCs	Title
RFC 4075	*Simple Network Time Protocol (SNTP) Configuration Option for DHCPv6*
RFC 4087	*IP Tunnel MIB*
RFC 4091	*The Alternative Network Address Types (ANAT) Semantics for the Session Description Protocol (SDP) Grouping Framework*
RFC 4092	*Usage of the Session Description Protocol (SDP) Alternative Network Address Types (ANAT) Semantics in the Session Initiation Protocol (SIP)*
RFC 4109	*Algorithms for Internet Key Exchange version 1 (IKEv1)*
RFC 4191	*Default Router Preferences and More-Specific Routes*
RFC 4193	*Unique Local IPv6 Unicast Addresses*
RFC 4214	*Intra-Site Automatic Tunnel Addressing Protocol (ISATAP)*
RFC 4242	*Information Refresh Time Option for Dynamic Host Configuration Protocol for IPv6 (DHCPv6)*
RFC 4282	*The Network Access Identifier*
RFC 4283	*Mobile Node Identifier Option for Mobile IPv6*
RFC 4285	*Authentication Protocol for Mobile IPv6*
RFC 4291	*IP Version 6 Addressing Architecture*
RFC 4292	*IP Forwarding Table MIB*
RFC 4293	*Management Information Base for the Internet Protocol (IP)*
RFC 4302	*IP Authentication Header*
RFC 4306	*Internet Key Exchange (IKEv2) Protocol*
RFC 4308	*Cryptographic Suites for IPsec*
RFC 4364	*BGP MPLS/IP Virtual Private Networks (VPNs)*
RFC 4382	*MPLS/BGP Layer 3 Virtual Private Network (VPN) Management Information Base*

RFCs	Title
RFC 4443	*Internet Control Message Protocol (ICMPv6) for the Internet Protocol Version 6 (IPv6) Specification*
RFC 4552	*Authentication/Confidentiality for OSPFv3*
RFC 4594	*Configuration Guidelines for DiffServ Service Classes*
RFC 4601	*Protocol Independent Multicast - Sparse Mode (PIM-SM): Protocol Specification*
RFC 4610	*Anycast-RP Using Protocol Independent Multicast (PIM)*
RFC 4649	*Dynamic Host Configuration Protocol for IPv6 (DHCPv6) Relay Agent Remote-ID Option*
RFC 4659	*BGP-MPLS IP Virtual Private Network (VPN) Extension for IPv6 VPN*
RFC 4724	*Graceful Restart Mechanism for BGP*
RFC 4798	*Connecting IPv6 Islands over IPv4 MPLS Using IPv6 Provider Edge Routers (6PE)*
RFC 4818	*RADIUS Delegated-IPv6-Prefix Attribute*
RFC 4861	*Neighbor Discovery for IP version 6 (IPv6)*
RFC 4862	*IPv6 Stateless Address Autoconfiguration*
RFC 4884	*Extended ICMP to Support Multi-Part Messages*
RFC 4885	*Network Mobility Support Terminology*
RFC 4887	*Network Mobility Home Network Models*
RFC 5015	*Bidirectional Protocol Independent Multicast (BIDIR-PIM)*
RFC 5059	*Bootstrap Router (BSR) Mechanism for Protocol Independent Multicast (PIM)*
RFC 5072	*IPv6 over PPP*
RFC 5095	*Deprecation of Type 0 Routing Headers in IPv6*
RFC 5120	*M-ISIS: Multi Topology (MT) Routing in Intermediate System to Intermediate Systems (IS-ISs)*

RFCs	Title
RFC 5130	*A Policy Control Mechanism in IS-IS Using Administrative Tags*
RFC 5187	*OSPFv3 Graceful Restart*
RFC 5213	*Proxy Mobile IPv6*
RFC 5308	*Routing IPv6 with IS-IS*
RFC 5340	*OSPF for IPv6*
RFC 5460	*DHCPv6 Bulk Leasequery*
RFC 5643	*Management Information Base for OSPFv3*
RFC 5838	*Support of Address Families in OSPFv3*
RFC 5844	*IPv4 Support for Proxy Mobile IPv6*
RFC 5845	*Generic Routing Encapsulation (GRE) Key Option for Proxy Mobile IPv6*
RFC 5846	*Binding Revocation for IPv6 Mobility*
RFC 5881	*Bidirectional Forwarding Detection (BFD) for IPv4 and IPv6 (Single Hop)*
RFC 5905	*Network Time Protocol Version 4: Protocol and Algorithms Specification*
RFC 5969	*IPv6 Rapid Deployment on IPv4 Infrastructures (6RD) -- Protocol Specification*
RFC 6105	*IPv6 Router Advertisement Guard*