Chapter 15

Internet Protocol Version 6 (IPv6)

Introduction ................................................................. 15-3
Overview of IPv6 .......................................................... 15-3
The 6bone ................................................................. 15-4
IPv6 Addresses and Prefixes ........................................... 15-4
IPv6 Headers ............................................................ 15-6
The Internet Control Message Protocol (ICMPv6) .......... 15-8
IPv6 Routing ............................................................. 15-11
IPv6 Filtering .......................................................... 15-11
Integration of IPv4 and IPv6 ......................................... 15-12
IPv6 on the Router .................................................... 15-12
Enabling IPv6 .......................................................... 15-13
IPv6 Interfaces and Addresses ..................................... 15-13
Extension Header Processing ...................................... 15-15
Routing Table Processing and RIPv6 ......................... 15-15
Neighbour Discovery .................................................. 15-16
IPv6 Filtering .......................................................... 15-17
IPv6 Fragmentation ................................................... 15-18
Telnet v6 ................................................................. 15-18
Ping .......................................................................... 15-19
Secure Shell ............................................................. 15-20
Tunnelling IPv6 packets over IPv4 ................................. 15-21
6-to-4 ................................................................. 15-21
Static Tunnelling ..................................................... 15-22
Configuration Examples .............................................. 15-23
Basic Routing .......................................................... 15-23
Dynamic Routing with RIPv6 ...................................... 15-24
Dynamic (6-to-4) Tunnelling over an IPv4 Network ....... 15-28
Static Tunnelling over an IPv4 Network ....................... 15-30
IPv6 Filters ............................................................. 15-32
Command Reference ................................................. 15-34
add ipv6 6to4 ........................................................... 15-34
add ipv6 filter .......................................................... 15-35
add ipv6 host ............................................................ 15-40
add ipv6 interface ..................................................... 15-41
add ipv6 nd ............................................................. 15-43
add ipv6 prefix ........................................................ 15-44
add ipv6 rip ............................................................. 15-45
add ipv6 route ......................................................... 15-46
add ipv6 tunnel ......................................................... 15-48
create ipv6 interface ................................................. 15-49
delete ipv6 6to4 ......................................................... 15-50
delete ipv6 filter ................................................................. 15-50
delete ipv6 host ............................................................... 15-51
delete ipv6 interface ...................................................... 15-51
delete ipv6 nd ................................................................. 15-52
delete ipv6 prefix ........................................................... 15-53
delete ipv6 rip ................................................................. 15-53
delete ipv6 route ............................................................ 15-54
delete ipv6 tunnel .......................................................... 15-55
destroy ipv6 interface ...................................................... 15-55
disable ipv6 ................................................................. 15-56
disable ipv6 advertise ..................................................... 15-56
disable ipv6 debug ......................................................... 15-57
disable ipv6 mtudiscovery .............................................. 15-57
disable ipv6 rip .............................................................. 15-58
enable ipv6 ................................................................. 15-58
enable ipv6 advertise ..................................................... 15-58
enable ipv6 debug ......................................................... 15-59
enable ipv6 mtudiscovery .............................................. 15-59
enable ipv6 rip .............................................................. 15-59
reset ipv6 ndcache ........................................................ 15-60
set ipv6 filter ............................................................... 15-61
set ipv6 interface ........................................................ 15-64
set ipv6 mtu ................................................................. 15-65
set ipv6 nd ................................................................. 15-66
set ipv6 prefix ............................................................ 15-68
set ipv6 route preference ............................................. 15-68
show ipv6 ................................................................. 15-69
show ipv6 counter ........................................................ 15-70
show ipv6 filter ........................................................... 15-72
show ipv6 host ............................................................ 15-76
show ipv6 interface ...................................................... 15-78
show ipv6 multicast ...................................................... 15-79
show ipv6 ndcache ........................................................ 15-81
show ipv6 ndconfig ....................................................... 15-82
show ipv6 rip .............................................................. 15-83
show ipv6 route ............................................................ 15-85
show ipv6 route multicast ............................................. 15-87
show ipv6 route preference ......................................... 15-88
show ipv6 tunnel .......................................................... 15-90
Introduction

This chapter describes the main features of IPv6, the router’s implementation of IPv6 and how to configure and operate IPv6 on the router.

This chapter describes the following IPv6 features:
- Linking together networks that run IPv6.
- Allowing address autoconfiguration of hosts connected to the router.
- Enabling IPv6 to operate over existing IPv4-based networks.

For more information about multicast support for IPv6 on the router, see Chapter 18, IPv6 Multicasting.

For more information about configuring security features for IPv6, see Chapter 45, IP Security (IPsec) and Chapter 25, Compression and Encryption Services.

IPv6 is enabled with a special feature licence. To obtain a special feature licence contact an Allied Telesyn authorised distributor or reseller.

Overview of IPv6

IPv6 is the next generation of the Internet Protocol (IP). It has primarily been developed to solve the problem of the eventual exhaustion of the IPv4 address space, but also offers other enhancements. IPv6 addresses are 16 bytes long, in contrast to IPv4’s 4 byte addresses. Other features of IPv6 include:

- Address structure improvements:
  - globally unique addresses with more levels of addressing hierarchy to reduce the size of routing tables
  - autoconfiguration of addresses by hosts
  - improved scalability of multicast routing by adding a “scope” field to multicast addresses
  - a new type of address, the “anycast address”, which sends a packet to any one of a group of devices

- Removal of the need for packet fragmentation en-route by dynamic determination of the largest packet size that is supported by every link in the path. A link’s MTU (Maximum Transmission Unit) must be at least 1280 bytes, compared with 576 bytes for IPv4.

- Traffic Class, which allows a packet to be labelled with an appropriate priority. If the network becomes congested, the lowest priority packets are dropped.

- Flow labels, which indicate to intermediate routers that packets are part of a flow, and that this flow requires a particular type of service. This feature enables, for example, real-time processing of data streams. It also increases routing speed because the forwarding router need only check the flow label, not the rest of the header. The handling indicated by the flow label can be done by the IPv6 Hop-by-Hop header, or by a separate protocol such as RSVP.

- Mandatory authentication and data integrity protocols through IPsec. IPsec is optional in IPv4.
An IPv6 network can contain three types of nodes, where *node* is a general term that refers to any IPv6-aware device. A *host* is a device on the network that is not a router. For example, a host may be a printer or a computer. A router may also act as a host. A *router* is a device on the network that directs the flow of IPv6 packets. For example, a router may be a router or a Layer 3 switch. A *destination* is a host to which packets are specifically sent.

**The 6bone**

The 6bone is an experimental virtual network of nodes that support IPv6 packets, tunneled together through the existing IPv4 Internet. Most of the nodes are workstations or similar machines, with IPv6-capable operating systems. The theory of tunnelling IPv6 packets over an IPv4 network is outlined in “Integration of IPv4 and IPv6” on page 15-12.

The 6bone is part of the transition to IPv6. Its purpose is to provide an environment in which IPv6 can be tested and procedures for IPv6 can be developed. When IPv6 is sufficiently developed and being used widely, the 6bone will probably disappear.

**IPv6 Addresses and Prefixes**

IPv6 addresses are hexadecimal, and are made up of eight pairs of octets separated by colons. An example of a valid address is `fe80:0000:0000:0000:0260:0000:97ff:64aa`. In the interests of brevity, addresses can be abbreviated in two ways:

- Leading zeros can be omitted, so this address can be written as `fe80:0:0:0:260:0:97ff:64aa`.
- Consecutive zeros can be replaced with a double colon, so this address can be written as `fe80::260:0:97ff:64aa`. Note that a double colon can replace any number of consecutive zeros, but an address can contain only one double colon.

Like IPv4 addresses, a proportion of the leftmost bits of the IPv6 address can be used to indicate the subnet, rather than a single node. This part of the address is called the *prefix*. Prefixes provide the equivalent functionality to a subnet mask in IPv4, allowing a subnet to be addressed, rather than a single node. If a prefix is specified, the IPv6 address is followed by a slash and the number of bits that represent the prefix. For example, `3ffe::/16` indicates that the first 16 bits (3ffe) of the address `3ffe:0:0:0:0:0:0:0` represent the prefix.

Like IPv4 addresses, IPv6 addresses are attached to interfaces.

**Unicast Addresses**

A unicast address is attached to a single interface and delivers packets only to that interface.

The following special addresses have been defined:

- IPv4-compatible and IPv4-mapped addresses. IPv4-compatible addresses are used to tunnel IPv6 packets across an IPv4 network. IPv4-mapped addresses are used by an IPv6 host to communicate with an IPv4 host. The IPv6 host addresses the packet to the mapped address.
Link-local addresses, which can be used on the local network that the interface is attached to. The link-local prefix is fe80::/10. Different interfaces on a device may have the same link-local address.

Site-local addresses, which are used in internal or private networks. These addresses are analogous to the IPv4 private addresses 10.x.x.x and 192.168.x.x.

The Loopback address, consisting of ::1, which is the equivalent of the IPv4 loopback address, and allows a host to send packets to itself.

The Unspecified address, consisting of ::, which is the equivalent of the IPv4 unspecified address, and is used as a source address by hosts during the autoconfiguration process.

Multicast Addresses
For information about IPv6 multicast group management and multicast routing, see Chapter 18, IPv6 Multicasting.

IPv6 multicast addresses also provide the equivalent functionality to broadcast addresses in IPv4. Broadcast addresses are not supported in IPv6. A multicast address identifies a group of interfaces, and packets are sent to all interfaces in that group.

Among the special addresses that have been defined are addresses that allow multicasting to:

- All interfaces on a particular host (FF01::1)
- All nodes on a local network (FF01::2)
- All routers on the local link (FF02::2)
- All routers on the local site (FF05::2).

Anycast Addresses
An anycast address is a unicast address that is attached to more than one interface. If a packet is sent to an anycast address it is delivered to the nearest interface with that address, with the definition of “nearest” depending on the protocol used for routing. If the protocol is RIPv6, the nearest interface is the one that is the shortest number of hops away.

Anycast addresses can be assigned to routers only, and packets cannot originate from an anycast address. A router must be configured to know that it is using an anycast address because the address format cannot be distinguished from that of a unicast address.

Only one anycast address has been predefined: the subnet-router address. The subnet-router address sends messages to the nearest router on a subnet and consists of the subnet’s prefix followed by zeros.
**IPv6 Headers**

The basic unit of data sent through an internet is called a *packet* in IPv6. A packet consists of a *header* followed by the *data* (Figure 15-1 on page 15-6, Table 15-1 on page 15-7). The header contains the information necessary to move the packet across the internet. It must be able to cope with missing and duplicated packets as well as possible fragmentation (and reassembly) of the original packet.

The IPv6 header is similar to the shorter IPv4 header, which is described in Figure 14-1 and Table 14-1 on page 14-7 of Chapter 14, Internet Protocol (IP). The IPv6 header is twice as long as the IPv4 header (40 bytes instead of 20 bytes) and contains four times the address space size (128 bits instead of 32 bits).

**The Basic IPv6 Header**

The IPv6 header no longer contains the header length, identification, flags, fragment offset, and header checksum fields. Some of these options are placed in extension headers. The Time To Live field is replaced with a hop limit, and the IPv4 Type of Service field is replaced with a Differentiated Services field. The Differentiated Services field contains the DSCP bits, used in a Quality of Service (QoS) regime.

Figure 15-1: The IPv6 packet

<table>
<thead>
<tr>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>0</th>
<th>1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ver</td>
<td>Diff Serv</td>
<td>Flow Label</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Payload Length</td>
<td>Next Header</td>
<td>Hop Limit</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

- 128-bit source IP address
- 128-bit destination IP address
- Optional extension headers
- User data
The IPv4 header fields that have been changed in IPv6 are:

- **Type of Service**
  
The type of service that a connection should receive is indicated in IPv6 by the Flow Label field in the IPv6 header.

- **Fragmentation information (the Identification field, the Flags field and the Fragment Offset field)**
  
  In most cases fragmentation does not occur in IPv6. If it does, packets are fragmented at their source and not en route. Therefore, the fragmentation information is contained in an extension header to reduce the size of the basic IPv6 header.

- **Header Checksum**
  
  This option has not been provided in IPv6. This is because transport protocols implement checksums and because of the availability of the IPsec authentication header (AH) in IPv6.
Options

Extension headers handle all the optional values associated with IPv6 packets. The biggest advantage of this scheme is that the size of the basic IP header is a constant.

Extension Headers

In IPv6 many of the less commonly used fields in the IPv4 header (or their equivalents) have become extension headers, which are placed after the basic IPv6 header. The length of each header must be a multiple of 8 bytes.

The first extension header is identified by the Next Header field in the basic IPv6 header. Any subsequent extension headers are identified by an 8-bit "Next Header" value at the beginning of the preceding extension header.

IPv6 nodes that originate packets are required to place extension headers in a specific order:

1. The basic IPv6 header, which must come immediately before the extension headers.
2. The Hop-by-Hop header, which specifies options that must be examined by every node in the routing path.
3. A Destination Options header, to specify options to be processed by the first destination or final destination. The destination options header is the only extension header that may be present more than once in the IPv6 packet.
4. The Routing header, which allows a static path to be specified for the packet, if the dynamically-determined path is undesirable.
5. The Fragment header, which indicates that the source node has fragmented the packet, and contains information about the fragmentation.
6. The Authentication header (AH), which verifies the integrity of the packet and its headers. The AH is an IPsec feature, and is described in "IP Security (IPsec)" on page 45-4 of Chapter 45, IP Security (IPsec).
7. The Encapsulating Security Payload (ESP) header, which encrypts a packet and verifies the integrity of its contents. The ESP is an IPsec feature, and is described in "IP Security (IPsec)" on page 45-4 of Chapter 45, IP Security (IPsec).
8. The Upper Layer Protocol header, which indicates which protocol a higher layer (such as the transport layer) is to process the packet with (for example, TCP).

The Internet Control Message Protocol (ICMPv6)

The Internet Control Message Protocol, ICMPv6, provides a mechanism for error reporting and route discovery and diagnostics. It also conveys information about multicast group membership, a function that is carried out by the Internet Group Management Protocol (IGMP) in IPv4, and performs address resolution, which the Address Resolution Protocol (ARP) performs in IPv4.

Significant aspects of ICMPv6 include neighbour discovery, which allows one device in a network to find out about other nearby devices, and stateless address autoconfiguration, which allows a device to dynamically determine its own IPv6 address.
ICMPv6 is also used to support the Ping v6 (Packet Internet Groper) and Trace route v6 functions, which are used to verify the connections between networks and network devices. Ping is used to test the connectivity between two network devices to determine whether each network device can “see” the other device. Trace route is used to discover the route used to pass packets between two systems running the IP protocol.

Both of these functions operate almost identically in IPv4 and IPv6. For more information, see “Ping and Trace Route” on page 14-38 of Chapter 14, Internet Protocol (IP).

**Neighbour Discovery**

Neighbour discovery is an ICMPv6 function that allows a router or host to identify other devices on its links. This information is then used in address autoconfiguration, to redirect a node to use a more appropriate router if necessary, and to maintain reachability information with its neighbours.

The IPv6 Neighbour Discovery protocol is similar to a combination of the IPv4 protocols ARP, ICMP Router Discovery and ICMP Redirect.

The packet types involved with neighbour discovery are:

- **Router solicitation**, in which a host sends out a request for routers to generate advertisements.
- **Router advertisement**, which allows routers to advertise their presence and other network parameters. A router sends an advertisement packet in response to a solicitation packet from a host.
- **Neighbour solicitation**, in which a node sends a packet to determine the link layer address of a neighbour or to verify that a neighbour is still active.
- **Neighbour advertisement**, which is a response to a neighbour solicitation packet. These packets are also used to notify neighbours of link layer address changes.
- **Redirect**, which is used to inform hosts of a better first hop.

To comply with Section 6.2.1 of RFC 2461, IPv6 Neighbour Discovery, the router does not generate router advertisements by default. See “Neighbour Discovery” on page 15-16 for instructions about enabling advertisements.

The packet types provide the following services:

- **Address resolution**
  
  This is a method for carrying out address autoconfiguration, and is achieved using the Neighbour Solicitation Message and the Neighbour Advertisement Message.

- **Router and prefix discovery**
  
  On connection to a link, a node needs to know the address of a router that the node can use to reach the rest of the world. The node also needs to know the prefix (or prefixes) that define the range of IP addresses on its link that it can reach without going through a router.

  Routers use ICMP to convey this information to hosts, by means of router advertisements. The message may have an option attached (the source link address option), which enables the receiving node to respond directly to the router, without performing a neighbour solicitation.
■ Immediate information

The configuration of a router includes a defined frequency at which unsolicited advertisements are sent. If a node wants to obtain information about the nearest router immediately, rather than waiting for the next unsolicited advertisement, the node can send a router solicitation message. Each router that receives the solicitation message sends a router advertisement specifically to the node that sent the solicitation.

■ Redirection

If a node is aware of more than one router that it can use to connect to wider networks, the router to which it sends packets by default does not always represent the most desirable route. ICMPv6 uses the redirect packet to communicate a more effective path to the node.

■ Neighbour Unreachability Detection (NUD)

A node may issue solicitation requests to determine whether a path is still viable, or may listen in on acknowledgement packets of higher-layer protocols, such as TCP. If the node determines that a path is no longer viable, it attempts to establish a new link to the neighbour, or to re-establish the previous link. NUD can be used between any two devices in the network, independent of whether the devices are acting as hosts or routers.

Stateless address autoconfiguration

Stateless address autoconfiguration allows an IPv6-aware device to be plugged into a network without manual configuration with an IP address. This plug and play functionality results in networks that are easier to set up and modify, and simplifies the process of shifting to use a new Internet Service Provider (ISP).

Stateless address autoconfiguration is achieved in a series of steps. The first three steps autoconfigure a link-local address, and the last three a global address. The first three steps are performed by routers, and all six steps are performed by hosts.

On the router or host:

1. During system start-up, the node begins autoconfiguration by generating a link-local address for the interface. A link-local address is formed by adding the interface ID to the link-local prefix fe80::/10.

2. The node then transmits a neighbour solicitation message to this address. If the address is already in use, the node that the address belongs to replies with a neighbour advertisement message. The autoconfiguration process stops and manual configuration of the node is then required.

3. If no neighbour advertisement is received, the node concludes that the address is available and assigns it to the chosen interface.

On the host:

4. The node then sends one or more router solicitations to detect if any routers are present. Any routers present responds with a router advertisement.

   If no router advertisement is received, the node tries to use DHCP to obtain an address and other configuration information. If no DHCP server responds, the node continues using the link-level address.

   If a router advertisement is received, this message informs the node how to proceed with the auto configuration process.
5. The prefix from the router advertisement, if received, is added to the link-level address to form the global unicast IP address.

6. This address is then assigned to the network interface.

If routers are present, the node continues to receive router advertisements. The node updates its configuration when there are changes in the router advertisements.

**IPv6 Routing**

Routing in IPv6 is almost identical to IPv4 routing under CIDR, except that the addresses are 128-bit IPv6 addresses instead of 32-bit IPv4 addresses. More information about routing can be found in “Routing” on page 14-17 of Chapter 14, Internet Protocol (IP).

**Routing Information Protocol, RIPv6**

RIP is a simple distance vector protocol that defines networks based on how many hops they are from the router. When a network is more than 15 hops away (one hop is one link), it is not included in the routing table.

RIPv6, also referred to as RIPvng (for “next generation”) is similar to RIPv2, which is described in “RIP” on page 14-24 of Chapter 14, Internet Protocol (IP). Extensions to RIPv2 to support IPv6 are:

- The address field of a routing entry is expanded to 128 bits to allow IPv6 prefixes
- The 32-bit RIPv2 subnet mask field is replaced by an 8-bit prefix length field
- Authentication is removed in RIPv6
- The size of a routing packet is no longer arbitrarily limited
- RIPv6 specifies the next hop instead of simply allowing the recipient of the update to set the next hop to the sender of the update.

In RIPv6, each router uses a routing table to keep track of every destination that is reachable throughout the system. Each entry in the routing table contains:

- The IPv6 prefix of the destination
- A metric, which represents the total cost of getting a packet from the router to that destination
- The IPv6 address of the next router along the path to the destination
- A flag to indicate that information about the route has changed recently
- Various timers associated with the route.

**IPv6 Filtering**

With the increase in connections to the Internet, and the interconnection of networks from different organisations, filtering data packets is an important way to ensure that only legitimate connections are allowed. Security can never be perfect while connections to other networks exist, but filters let network managers manage permissible access while restricting users without permission.
IPv6 filtering uses a similar mechanism to IPv4 filtering. More information can be found in “Policy-Based Routing” on page 14-27 of Chapter 14, Internet Protocol (IP).

Integration of IPv4 and IPv6

IPv6 has been designed in such a way that a smooth transition from IPv4 is possible. The most effective way to ensure this is to use a dual IP stack. A node configured as a dual stack system has both a 128-bit IPv6 address and a 32-bit IPv4 address, and can communicate with nodes running only IPv4 and nodes running only IPv6.

Another aspect of the transition period is the tunnelling of IPv6 packets across the IPv4 network. IPv6 packets are tunnelled simply by encapsulating the IPv6 packet within an IPv4 datagram, and identifying that this datagram is an encapsulated IPv6 packet by giving the datagram a protocol value of 41.

IPv6 on the Router

This section describes the router’s support for IPv6, and how to configure IPv6 on the router. Fundamental IPv6 features on the router are:

- IPv6 interfaces and addresses
- Extension header processing
- Routing table processing
- RIPv6 (RIPvng)
- Neighbour discovery
- Stateless Address Autoconfiguration
- IPv6 filtering
- IPv6 fragmentation
- IPv6 multicasting, which is described in Chapter 18, IPv6 Multicasting
- IPsec and ISAKMP, which are described in Chapter 45, IP Security (IPsec)
- Dynamic Host Configuration Protocol (DHCP6), which is described in Chapter 36, Dynamic Host Configuration Protocol for IPv6 (DHCP6)

The router also supports the following upper layer protocols:

- UDP, which transports RIPv6 packets
- TCP, which transports Telnet requests
- ICMPv6, which is used for stateless address autoconfiguration, neighbour discovery, Ping and Trace Route requests.

Integration of IPv6 with IPv4 is provided by:

- 6-to-4 support
- IPv6 static tunnelling.

Many of these features are performed automatically by the router, and most commands operate in a similar manner to their IPv4 equivalents.
Enabling IPv6

The router’s implementation of IPv6 is disabled by default. To enable IPv6, use the command:

```
enable ipv6
```

To disable IPv6, use the command:

```
disable ipv6
```

Any IPv6 configuration that the router has performed dynamically is preserved between disabling and re-enabling IPv6. For example, addresses that have been configured are still present.

To display information about IPv6 settings, use the command:

```
show ipv6
```

To display IPv6 counters, use the command:

```
show ipv6 counter
```

Because the router implements IPv6 as a dual stack, implementing IPv6 does not affect IPv4 functionality.

IPv6 Interfaces and Addresses

The router supports the addition of IPv6 addresses directly to Point-to-Point Protocol (PPP) and physical Ethernet (eth) and Virtual Local Area Network (VLAN) interfaces and indirectly to virtual interfaces, when the tunnel is created.

As with IPv4 addresses, a proportion of the leftmost bits of the IPv6 address can be the address of a subnet, rather than a host. This subnet part of the address is called the *prefix*. The prefix is specified by following the IPv6 address with a slash (/) and the length of the prefix in bits:

```
ipv6address=ipv6add/prefix-length
```

This syntax is referred to as *slash notation*.

To create an IPv6 logical interface and associate it with an interface, use the command:

```
create ipv6 interface=interface [duptrans=1..16] [retrans=0..4294967295]
```

where *interface* is an interface name formed by concatenating an interface and an interface instance (e.g. ppp1, eth, vlan1).

As part of the creation process, the router performs stateless address autoconfiguration to assign an IPv6 address to the interface, by adding the interface’s MAC address to the reserved IPv6 prefix fe80::. These addresses are link-local addresses, which are sufficient for communication among devices on the same link. Several interfaces can be given the same link-local address, as specified in RFC 2373 "IP Version 6 Addressing Architecture".

The DUPTRANS parameter sets the number of neighbour solicitation messages that the router sends during the Duplicate Address Detection process. The RETRANS parameter sets the number of times the router resends each Router
Advertisement message. Lossy links can be partially compensated for by increasing these counters.

If the router is used as a host instead of a router, it is still able to use stateless address autoconfiguration to generate a link-local address. However, it is unable to receive router advertisements, and therefore unable to autoconfigure a global address. If it is necessary to give the router a global IPv6 address, it must be added manually.

To add another IPv6 address manually to the interface, or to create the interface and manually add the IPv6 address to it at the same time, use the command:

```
add ipv6 interface=interface
ipaddress=ipv6add/prefix-length [filter=0..99]
[preferred=1..4294967295|infinite]
[priorityfilter=200..299] [publish={yes|no}]
[type={anycast|unicast}] [valid=1..4294967295|infinite]
```

An interface can have only one link-local address at a time. If a new link-local address is added on an IPv6 interface, the interface uses the new link-local address. The link-local address of a static tunnel cannot be changed after the tunnel has been created (see “Static Tunnelling” on page 15-22 for information about tunnels). The link-local address of an interface cannot be changed while PIM6 is attached to it. The PIM6 interface must first be deleted by using the command:

```
delete pim6 interface in Chapter 18, IPv6 Multicasting
```

If the address is an anycast address (see “Anycast Addresses” on page 15-5), the TYPE parameter allows it to be distinguished from a unicast address.

To change the address or other parameters of the interface, use the command:

```
set ipv6 interface=interface
ipaddress=ipv6add/prefix-length [filter=0..99]
[preferred=1..4294967295|infinite]
[priorityfilter=200..299] [publish={yes|no}]
[type={anycast|unicast}] [valid=1..4294967295|infinite]
```

To destroy an IPv6 interface, use the command:

```
destroy ipv6 interface=interface
```

To delete an address from an interface, use the command:

```
delete ipv6 interface=interface ipaddress=ipv6add
```

Deleting a link-local address on an interface causes the interface to revert back to its original auto-configured link-local address. An interface’s auto-configured IPv6 link-local address cannot be deleted except by destroying the interface.

To display information about the configured interfaces, use the command:

```
show ipv6 interface[=interface]
```

To display information about IPv6 multicast addresses, use the command:

```
show ipv6 multicast
```

To associate an IPv6 address with the name of a host, use the command:

```
add ipv6 host=name ipaddress=ipv6add
```
This functionality is similar to the `add ip host` command on page 14-76 of Chapter 14, Internet Protocol (IP).

To disassociate the IPv6 address and the host name, use the command:
```
delete ipv6 host=name
```

To display information about the host name table, use the command:
```
show ipv6 host
```

**Extension Header Processing**

All routers in the path to the final destination process the routing header and the hop by hop header to route the packet to the specified path. The final node processes the fragment header. When the router is the source or final destination of the packet, it processes these extension headers as required.

**Routing Table Processing and RIPv6**

The router maintains and processes a routing table for IPv6 addresses, in a similar manner to the IPv4 routing table described in “Routing” on page 14-17 of Chapter 14, Internet Protocol (IP).

The Router Information Protocol (RIPv6 or RIPng, as described in RFC 2080, “RIPng for IPv6”) is supported, and allows the router to share information from its routing tables with routers that connect other networks. RIPv6 passes routing table information from neighbour to neighbour along a line of routers.

RIPv6 packets are transported over UDP. When IPv6 is enabled, the router can route UDP packets through an IPv6 network or tunnel.

RIPv6 routing is disabled by default. To enable it, use the command:
```
enable ipv6 rip
```

To enable receiving or sending of RIP packets on an interface, use the command:
```
add ipv6 rip interface=interface poisonreverse={on|off}
```

This command also specifies whether poison reverse is enabled on the interface. Poison reverse addresses the problem of slow convergence on RIPng routes. If one device in a network goes down, it can take a long time for the devices to recognise that routes through the crashed device are no longer available. With poison reverse, when a device goes down, the devices next to it continues to advertise the route but with a cost of 16. This cost indicates that the route is unavailable.

To stop an interface from sending or receiving RIP packets, use the command:
```
delete ipv6 rip interface=interface
```

To disable RIP, use the command:
```
disable ipv6 rip
```

To display information about RIP counters or timers, use the command:
```
show ipv6 rip [counter|timer]
```
Under some conditions, the route that is dynamically determined by RIPv6 may not be the most desirable one. For example, certain packets may need to be sent over a more secure route. To statically add the desired route to an interface, use the command:

```
add ipv6 route=ipv6add/prefix-length interface=interface
nexthop=ipv6add [metric=1..16] [preference=0..65535]
```

The ROUTE parameter identifies the final destination network to which the packets are sent. This command can also add a default route where packets can be sent when no other route is found, for example, the gateway between the LAN and the wider network. To add a default route, use the command:

```
add ipv6 route=::/0 interface=interface
nexthop=ipv6add-to-send-packets-to
[metric=1..16] [preference=0..65535]
```

To delete a route from an interface, use the command:

```
delete ipv6 route=ipv6add interface=interface
nexthop=ipv6add
```

To display information about the IPv6 routes on the router, use the command:

```
show ipv6 route
```

**Neighbour Discovery**

The router issues router advertisement messages in response to a router solicitation message from a host so the host can determine the router’s identity and availability. The router also sends neighbour solicitation messages to neighbouring nodes and responds to neighbour solicitation messages with a neighbour advertisement message. The address resolution mechanism and queue structure is similar to that used in ARP in IPv4. More information about neighbour discovery can be found in RFC 2461, "Neighbour Discovery for IPv6".

To comply with Section 6.2.1 of RFC 2461, *IPv6 Neighbour Discovery*, the router does not generate router advertisements by default. To enable the router to generate router advertisements for all interfaces, use the command:

```
enable ipv6 advertise
```

To enable the router to generate router advertisements for a particular interface, use the command:

```
enable ipv6 advertise interface=interface
```

For each IPv6 interface you want to have advertised, also specify that the interface's prefix should be included in advertisements, using one of the commands:

```
add ipv6 interface=interface ipaddress={ipv6add/prefix
length|dhcp|dhcptemp|pd} publish=yes
```

```
set ipv6 interface=interface publish=yes
```

If the router does not generate router advertisements, the router discovery and address autoconfiguration processes in the network may not work as expected.

To disable router advertisements, use the command:

```
disable ipv6 advertise [interface=interface]
```

Neighbour Discovery
Neighbour discovery and solicitation are part of the IPv6 protocol and cannot be disabled.

To set neighbour discovery and Router Advertisement counters and timers, use the command:

```
set ipv6 nd
```

```
interface=interface [duptrans=1..16] [hop=1..255]
[life=0..4..9000] [maxaingt=4..1800] [mconf={yes|no}]
[minaingt=3..1350] [mtu=1280..65535] [oconf={yes|no}]
[reach=0..3600000] [retrans=0..4294967295]
```

To add a prefix to Router Advertisements for a particular interface, without adding the prefix to the interface's IPv6 address, use the command:

```
add ipv6 prefix=ipv6add/prefix-length interface=interface
```

```
[autonomous={yes|no}] [onlink={yes|no}]
[preferred=1..4294967295|infinite] [valid=1..4294967295|infinite]
```

To modify a prefix in the Router Advertisement prefix list, use the command:

```
set ipv6 prefix=ipv6add/prefix-length interface=interface
```

```
[autonomous={yes|no}] [onlink={yes|no}]
[preferred=1..4294967295|infinite] [valid=1..4294967295|infinite]
```

To display information about the neighbours determined by neighbour discovery, use the command:

```
show ipv6 ndcache
```

To display information about the neighbour discovery parameters, and the list of prefixes that are included in Router Advertisements, use the command:

```
show ipv6 ndconfig
```

### IPv6 Filtering

IPv6 packets can be filtered on arrival at the router, with a traffic filter, or on transmission from the router, with a priority filter. Traffic filters can be used to determine whether an incoming packet is accepted or rejected. Priority filters apply a particular priority to the packets. Priorities range from 0 to 7, with 0 having the highest priority.

To add an IPv6 filter, use the command:

```
add ipv6 filter=filter-id
```

```
source=ipv6add/prefix-length [action={include|exclude}]
[priority=p0..p7] [destination=ipv6add/prefix-length]
[dport={port-name|port-id|any}] [entry=entry-number]
[icmptype={icmp-type-name|icmp-type-id|any}]
[icmpcode={icmp-code-name|icmp-code-id|any}]
[log={4..1950|dump|header|none}]
[options={yes|no|on|off|enable|disable}]
[protocol={protocol|any|egp|icmp|ospf|tcp|udp}]
[session={any|established|start}]
[size={size|any}]
```

The `add ipv6 filter` command can be used to add another entry to an existing filter, by identifying the filter by its filter-id and using the ENTRY parameter to indicate the desired entry number for the new entry. Entries with the lowest entry numbers are higher in the filter and are applied to traffic before higher-numbered entries.
To modify an existing filter entry, use the command:

```
set ipv6 filter=filter-id
entry=entry-number [source=ipv6add/prefix-length]
[destination=ipv6add/prefix-length] [dport=(port-name|
port-id|any)] [icmptype=(icmp-type-name|icmp-type-id|
any)] [icmpcode=(icmp-code-name|icmp-code-id|
any)] [priority=p0..p7]
```

To delete a filter or one of its entries, use the command:

```
delete ipv6 filter=filter-id ENTRY={entry-number|ALL}
```

To display information about existing filters, sorted by filter and entry numbers, use the command:

```
show ipv6 filter=filter-id
```

### IPv6 Fragmentation

Routers along a path send an “ICMP packet too big” reply when they receive a packet that is larger than the MTU for the link. It is then up to the host to fragment the packets.

When the router is acting as a host, it performs fragmentation as needed. If the packet size is greater than the MTU the packet is fragmented into packets that are sized in multiples of 8 bytes. Each of the new packets carries a fragmentation header. A packet with a fragmentation header is identified with the value 44 in the previous header’s Next Header field.

### Telnet v6

The `telnet` command allows remote access to a device’s command-line interface. It operates similarly in IPv4 and IPv6. For more information, see “Telnet” on page 21-7 of Chapter 21, Terminal Server.

To Telnet to an IPv6 interface with the address 1111::260:0:97ff:64aa, use the command:

```
telnet 1111::260:0:97ff:64aa
```

Telneting to a link-local address requires interface information as well as the address, because a single link-local address can belong to several interfaces. To Telnet to a link-local address, specify the interface out which the Telnet request is sent, as well as the address. This interface is the interface, on the router from which the Telnet request originates, that is connected to the required destination interface (Figure 15-2 on page 15-19). The Telnet syntax is:

```
telnet ipv6-address%interface
```

For example:

```
telnet fe80::7c27%eth0
```
Instead of entering the IPv6 address of the host’s interface, an easy-to-remember name can be associated with the host by using the command:

```
add ipv6 host=name ipaddress=ipv6add
```

Telnet messages are transported over TCP. When IPv6 is enabled, the router can route TCP packets through an IPv6 network or tunnel.

**Ping**

The router supports an extended `ping` command on page 14-129 of Chapter 14, Internet Protocol (IP), which allows the user to attempt to contact an IPv6 interface and record whether packets are received and the response time if they are. Defaults, including a default address, can be entered with the `set ping` command on page 14-163 of Chapter 14, Internet Protocol (IP).

To ping an IPv6 interface with the address 1111::260:0:97ff:64aa, use the command:

```
ping 1111::260:0:97ff:64aa
```

Pinging a link-local address requires additional information, because a single link-local address can belong to several interfaces. To ping a link-local address, specify the interface out which the ping request is sent, as well as the address. This interface is the interface, on the router from which the ping request originates, that is connected to the required destination interface (Figure 15-2 on page 15-19). The ping syntax is:

```
ping ipv6-address%interface
```

For example:

```
ping fe80::7c27%eth0
```

If a source IPv6 address is also specified with the SIPADDRESS parameter, the source address must be on the outgoing interface. The SIPADDRESS cannot be a link-local address.
Secure Shell

The `ssh` command on page 43-25 of Chapter 43, Secure Shell initiates a Secure Shell connection to the specified host.

A Secure Shell request to an IPv6 link-local address requires interface information as well as the address, because a single link-local address can belong to several interfaces. To SSH to a link-local address, specify the interface out which the SSH request is sent, as well as the address. This interface is the interface, on the router from which the SSH request originates, that is connected to the required destination interface, Figure 15-2 on page 15-19 in Chapter 15, Internet Protocol Version 6 (IPv6). For example:

```
ssh fe80:7c27%vlan1 user=Admin password=18Again
```

Trace Route

The `trace` command on page 14-225 of Chapter 14, Internet Protocol (IP) records the path to an IPv6 node.

To view the route to an IPv6 node, use the command:

```
trace ipv6add
```
Tunnelling IPv6 packets over IPv4

6-to-4

The router can automatically tunnel IPv6 packets over IPv4 interfaces, as described in RFC 3056 - Connection of IPv6 Domains via IPv4 Clouds. The interfaces must first be correctly configured with global IPv4 addresses, and routes must be set up to ensure the router can communicate with the router at the other end of the tunnel using IPv4.

To configure a 6-to-4 interface, use the command:

```
add ipv6 6to4 ip=ipv4add
```

where `ipv4add` is the global IPv4 address of the IPv4 interface that IPv6 packets are transmitted and received on.

This command creates a virtual tunnel interface, with an interface name of `virtl`, and assigns it a link-local IPv6 address. The first tunnel created is numbered `virt0`, and each succeeding tunnel is given the next available instance number. Each virtual interface behaves as a normal IPv6 interface. The interface is given a link-local IPv6 address with the prefix 2002::/16, of the form:

```
2002::<ipv4-address>::<ipv4-address>
```

For example, if a PPP interface with the address 203.109.0.1 is configured as a 6-to-4 address, the IPv6 address of the virt0 interface is:

```
2002:cb6d:0001::cb6d:0001
```

To forward the desired IPv6 packets through the tunnel, add a route by using the command:

```
add ipv6 route=host-ipv6add/prefix-length int=6to4-tunnel-interface nexthop=ipv6add
```

where `host-ipv6add` is the IPv6 address of the destination host or router, `ipv6add` is the link-local address of the 6-to-4 interface on the router at the other end of the tunnel, and `6to4-tunnel-interface` is the VIRT interface that the router created (such as `virt0`).

To display tunnel interface names, instances, IPv6 addresses, and other information for all tunnels that are configured on the router, use the command:

```
show ipv6 tunnel
```

To display information about a specific tunnel, use the command:

```
show ipv6 interface=tunnel-interface
```

To remove a 6-to-4 interface, use the command:

```
delete ipv6 6to4 IP=ipv4add
```

“Dynamic (6-to-4) Tunnelling over an IPv4 Network” on page 15-28 is an example of this configuration.

Duplicate Address Detection (DAD) is not performed on IPv6 tunnels.
Static Tunnelling

As well as 6-to-4 automatic tunnelling, IPv6 subnets can be linked over an IPv4 tunnel by configuring static tunnels. “Static Tunnelling over an IPv4 Network” on page 15-30 is an example of this configuration.

To link two IPv6 networks through an IPv4 tunnel, first create the tunnel on each router by using the command:

```
add ipv6 tunnel local=ipv4add target=ipv4add
[interface=tunnel-interface] [ipaddress=ipv6add]
```

This command creates a virtual tunnel interface, with an interface name of `virt`, and assigns it a link-local IPv6 address. To specify an interface instance for the tunnel, enter it using the INTERFACE parameter (for example, `virt5`). Otherwise, the first tunnel created is numbered `virt0`, and each succeeding tunnel is given the next available instance number. Each virtual interface behaves as a normal IPv6 interface. The LOCAL parameter is the IPv4 address of the interface through which packets are sent and received on the local router. The TARGET parameter is the IPv4 address of the remote router’s interface. These IPv4 interfaces can be PPP or Ethernet or VLAN interfaces.

The tunnel can be given an IPv6 address manually, by specifying the optional parameter IPADDRESS. If IPADDRESS is not specified, the link-local address assigned to the tunnel interface is:

```
fe80::<local-ipv4-address>
```

where `local-ipv4-address` is in hexadecimal. For example, if one end of a tunnel is created by using the command:

```
add ipv6 tunnel local=192.168.1.2 target=192.168.1.1
```

then the interface’s link-local address is `fe80::c0a8:0102`.

Once the tunnel has been created, add a route on each interface to direct traffic to the other IPv6 network through the tunnel by using the command:

```
add ipv6 route=ipv6add interface=tunnel-interface
```

where `ipv6add` is the address of the IPv6 network on the other router (not the IPv6 address of the tunnel), and `tunnel-interface` is the interface that the tunnel was given when it was created (e.g. `virt0`).

To display tunnel interface names, instances, IPv6 addresses, and other information for all tunnels that are configured on the router, use the command:

```
show ipv6 tunnel
```

To display information about a specific tunnel, use the command:

```
show ipv6 interface=tunnel-interface
```

To delete a tunnel from an interface, use the command:

```
delete ipv6 tunnel=ipv6add interface=tunnel-interface
```

The IPv6 address in the TUNNEL parameter of the `delete` command is the original link-local address of the tunnel (the address that the router assigned to the tunnel when it was created).
Configuration Examples

The following examples illustrate the steps required to configure IPv6 on the router. The first example shows how to configure basic IPv6 routing, the second shows how to configure dynamic routing with RIP and the third example demonstrates tunnelling IPv6 packets through an IPv4 network. The final example shows how to configure IPv6 filtering on the router.

Before an IPv6 address can be added to a VLAN, the VLAN must be created, using the `create vlan` command. For more information about this command or VLANs, see Chapter 2, Switching on the AR410 and Chapter 3, Switching on the AR440S, AR441S and AR450S.

Basic Routing

This example demonstrates configuring a router’s Ethernet interface with an IPv6 address, and enabling it to access local hosts and external hosts through a gateway device. The gateway device and any other routers that the router needs to communicate with must also be configured with appropriate IPv6 interfaces, addresses, and routes.

To configure basic IPv6 routing on the router

1. Enable the IPv6 module.
   Enable IPv6 by using the command:
   ```
   enable ipv6
   ```

2. Add a global IPv6 address to an interface.
   To manually add the IPv6 address 3FFE::1/32 to the interface eth0, use the command:
   ```
   add ipv6 interface=eth0 ipaddress=3FFE::1/32
   ```

3. Check the automatically-created route.
   When an interface is added, an interface route is automatically created. Check this route by displaying the routing information by using the command:
   ```
   show ipv6 route
   ```
   This command produces output similar to the following figure.

   ![Figure 15-3: Example output from the `show ipv6 route` command for a basic IPv6 network](image-url)
The router can now communicate with other hosts on the network with the same prefix. To test that this route is functional, use the `ping` command to access another node in the local network:

```
ping 3ffe::2
```

Replies from the node are echoed on screen.

4. **Add a default route to the local gateway.**

The local gateway is the device connecting the local network to other networks such as the Internet. The default route is the route to which a packet is sent when the router cannot find a route for it. To add a default route, set the ROUTE parameter to ::/0 and the NEXTHOP parameter to the address where the packets should be sent.

To add a default route, when the router is connected to the local LAN via the eth0 interface and the local gateway’s IPv6 address is 3FFE::2, use the command:

```
add ipv6 route=::/0 next=3ffe::2 interface=eth0
```

5. **Test this route with ping.**

Once a return route has been created from the network 5FFE::/32 to 3FFE::/32, test communications between the networks by using the command:

```
ping 5ffe::1
```

The router can now access all local hosts with the same prefix, and to any host that the gateway device has a route.

### Dynamic Routing with RIPv6

RIP works between routers connected to multiple networks. This example demonstrates configuring three routers’ Ethernet interfaces with IPv6 addresses, and enabling RIPv6 routing on them so the routers at each extreme end can route to networks to which the others have access.

**Figure 15-4: Example for dynamic routing**

To configure dynamic routing with RIPv6

1. **Enable the IPv6 module.**

   Enable IPv6 by using the following command on each router:

   ```
   enable ipv6
   ```
2. **Create the IPv6 interfaces and add global IPv6 addresses to each required interface.**

Manually add IPv6 addresses to Ethernet interfaces on each router. On router A, use the command:

```
add ipv6 interface=eth0 ipaddress=4ffe::1/32
```

On router B, use the commands:

```
add ipv6 interface=eth0 ipaddress=4ffe::2/32
```
```
add ipv6 interface=eth1 ipaddress=5ffe::2/32
```

On router C, use the command:

```
add ipv6 interface=eth0 ipaddress=5ffe::1/32
```

The router automatically assigns a link local address to each interface.

3. **Check the routes on each router.**

When the interfaces are added, interface routes are automatically created. Check these routes by displaying the routing information, using the following command on each router:

```
show ipv6 route
```

This command produces output similar to the following figures.

Figure 15-5: The output of the `show ipv6 route` command on router A, showing the interface route

```
IPV6 Routing Table Entries
Destination prefix ---> Next Hop
Int.  Age Policy Protocol Metric Pref Tunnel DLCI Flags
------------------------------------------------------------
4ffe::/32 ---> ::
eth0  no  0      interface 1      0    no     -
------------------------------------------------------------
Codes: P=publish, D=default, A=addrconf, S=stale, L=onlink
N=nonexthop, C=cache, F=flow, U=unknown
```

Figure 15-6: The output of the `show ipv6 route` command on router B, showing the interface routes

```
IPV6 Routing Table Entries
Destination prefix ---> Next Hop
Int.  Age Policy Protocol Metric Pref Tunnel DLCI Flags
------------------------------------------------------------
4ffe::/32 ---> ::
eth0 no  0      interface 1      0    no     -
------------------------------------------------------------
5ffe::/32 ---> ::
eth1 no  0      interface 1      0    no     -
------------------------------------------------------------
Codes: P=publish, D=default, A=addrconf, S=stale, L=onlink
N=nonexthop, C=cache, F=flow, U=unknown
```
4. **Enable RIP on each router and add a RIP interface.**

To enable RIP, use the following command on each router:

```
enable ipv6 rip
```

To create an RIP interface on the network between the routers, use the following commands:

- **On router A:**
  ```
  add ipv6 rip interface=eth0
  ```

- **On router B:**
  ```
  add ipv6 rip interface=eth0
  add ipv6 rip interface=eth1
  ```

- **On router C:**
  ```
  add ipv6 rip interface=eth0
  ```

The RIP updates take 30 seconds to propagate between the routers. After this time, router A should contain a route to router C, and router C a route to router A.

5. **Check the routes.**

On each router, display the routes to check that the routes appear on the opposite router by using the command:

```
show ipv6 route
```

The output on router A should be similar to the following figure.

```
IPV6 Routing Table Entries
Destination prefix  ---  Next Hop
Int.  Age Policy  Protocol  Metric  Pref Tunnel  DLCI  Flags
------------------------------------------------------------
5ffe::/32  ---  ::
  eth0  no 0  interface 1  0  no  -
------------------------------------------------------------
Codes: P=publish, D=default, A=addrconf, S=stale, L=onlink
       N=nonexthop, C=cache, F=flow, U=unknown
```
The output on router C should be similar to the following figure.

Figure 15-9: The output of the `show ipv6 route` command on router C

<table>
<thead>
<tr>
<th>IPV6 Routing Table Entries</th>
<th>Destination prefix --</th>
<th>Next Hop</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Int. Age Policy Protocol Metric Pref Tunnel DLCI Flags</td>
<td></td>
</tr>
<tr>
<td>----------------------------</td>
<td>----------------------------------------------------------</td>
<td></td>
</tr>
<tr>
<td>5ffe::/32 --</td>
<td>::</td>
<td>eth0 no 0 interface 1 0 no -</td>
</tr>
<tr>
<td>4ffe::/32 --</td>
<td>fe80::0200:cdff:fe00:a148</td>
<td>eth0 yes 0 ripng 2 100 no -</td>
</tr>
</tbody>
</table>

Codes: P=publish, D=default, A=addrconf, S=stale, L=onlink
N=nonexthop, C=cache, F=flow, U=unknown

Note that the next hop for the RIPv6 routes is the link-local address for the other router, not the 4FFE::/32 or 5FFE::/32 address. This allows routers to share routes even if they aren’t on the same logical network. In this example, fe80::0200:cdff:fe00:a14d is the link-local address of eth0 on router B and fe80::0200:cdff:fe00:a148 is the link-local address of eth1 on router B.

6. Test these routes with PING.

On router A, use the command:
```
ping 5ffe::1
```

On router C, use the command:
```
ping 4ffe::1
```

Each router can now communicate with the networks to which they previously did not have a route.
Dynamic (6-to-4) Tunnelling over an IPv4 Network

A 6-to-4 tunnel allows IPv6 packets to be routed between IPv6-aware nodes that are connected by an IPv4 network. The router automatically encapsulates IPv6 packets. This configuration is shown in the following figure.

Figure 15-10: Example of a dynamic tunnel
To configure a 6-to-4 tunnel

1. **Enable IP and add an interface to each router.**

   Details on configuring basic IPv4 routing can be found in “A Basic TCP/IP Setup” on page 14-53 of Chapter 14, Internet Protocol (IP).

   On router A, use the commands:
   ```
   enable ip
   add ip interface=eth0 ipaddress=203.109.0.1
   ```

   On router B, use the commands:
   ```
   enable ip
   add ip interface=eth0 ipaddress=204.109.0.1
   ```

2. **Set up IPv4 routes to connect the routers.**

   On router A, use the command:
   ```
   add ip route=0.0.0.0 interface=eth0 nexthop=204.109.0.1
   ```

   On router B, use the command:
   ```
   add IP route=0.0.0.0 interface=eth0 nexthop=203.109.0.1
   ```

3. **Enable IPv6 on both routers.**

   Once the IPv4 connection is working correctly, enable IPv6 on each router by using the command:
   ```
   enable ipv6
   ```

4. **Configure the 6-to-4 interfaces.**

   On router A, use the command:
   ```
   add ipv6 6to4 ip=203.109.0.1
   ```

   The interface’s name is virt0. It has a link-local IPv6 address of 2002:cb6d:1::cb6d:1

   On router B, use the command:
   ```
   add ipv6 6to4 ip=204.109.0.1
   ```

   The interface’s name is virt0. It has a link-local IPv6 address of 2002:cc6d:1::cc6d:1

5. **Set up routes to forward packets via the tunnel.**

   On router A, use the command:
   ```
   add ipv6 route=::/0 interface=virt0
   nexthop=2002:cc6d:1::cc6d:1
   ```

   On router B, use the command:
   ```
   add ipv6 route=::/0 interface=virt0
   nexthop=2002:cb6d:1::cb6d:1
   ```

6. **Check the configuration.**

   Use the commands:
   ```
   show ipv6 interface=virt0
   show ipv6 tunnel
   ```
The output for router A is shown in the following figures.

Figure 15-11: Output of the `show ipv6 interface=virt0` command for router A

<table>
<thead>
<tr>
<th>IPV6 Interface Configuration</th>
</tr>
</thead>
<tbody>
<tr>
<td>Interface ..................... virt0</td>
</tr>
<tr>
<td>Ipv6 Interface Index .......... 3</td>
</tr>
<tr>
<td>Link-layer address .......... 6-to-4 interface</td>
</tr>
<tr>
<td>EUI-64 Interface Identifier .. 6-to-4 interface</td>
</tr>
<tr>
<td>IPSec ........................ No</td>
</tr>
<tr>
<td>True MTU/Link MTU .......... ~/1280</td>
</tr>
<tr>
<td>Multicast status ............ Enabled</td>
</tr>
<tr>
<td>Send Router Advertisements  No</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Ipv6 Interface Addresses :</th>
</tr>
</thead>
<tbody>
<tr>
<td>Int</td>
</tr>
<tr>
<td>-----</td>
</tr>
<tr>
<td></td>
</tr>
</tbody>
</table>

Figure 15-12: Output of the `show ipv6 tunnel` command for router A

<table>
<thead>
<tr>
<th>Ipv6 Tunnels:</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ipv6 Tunnel Address</td>
</tr>
<tr>
<td>Tunnel start</td>
</tr>
<tr>
<td>------------------</td>
</tr>
<tr>
<td>1 2002:cb6d:0001::cb6d:0001</td>
</tr>
</tbody>
</table>

**Static Tunnelling over an IPv4 Network**

A static tunnel allows IPv6 packets to be routed between IPv6-aware nodes that are connected by an IPv4 network. The addresses used in this example are shown in Table 15-2, and the configuration is shown in the following figure.
To configure a static tunnel

1. **Enable IP and add an interface to each router.**

   Details on configuring basic IPv4 routing can be found in “A Basic TCP/IP Setup” on page 14-53 of Chapter 14, Internet Protocol (IP).

   On router A, use the commands:
   ```
   enable ip
   add ip interface=eth0 ipaddress=192.168.1.1
   ```

   On router B, use the commands:
   ```
   enable ip
   add ip interface=eth0 ipaddress=192.168.1.2
   ```

2. **Enable IPv6 on both routers.**

   Once the IPv4 connection is working correctly, enable IPv6 on each router by using the command:
   ```
   enable ipv6
   ```

3. **Create a tunnel between the two routers.**

   On router A, use the command:
   ```
   add ipv6 tunnel local=192.168.1.1 target=192.168.1.2
   ```
   This command creates an IPv6 interface (virt0 if this is the first tunnel created), with link-local address fe80::c0a8:0101.

   On router B, use the command:
   ```
   add IPv6 tunnel local=192.168.1.2 target=192.168.1.1
   ```
   This command creates an IPv6 interface (virt0 if this is the first tunnel created), with link-local address fe80::c0a8:0102.

   The LOCAL parameter is the IPv4 address of the router at the local end of the tunnel, so for router A the LOCAL parameter is A’s IPv4 address and the TARGET parameter is B’s IPv4 address. For router B the LOCAL parameter is B’s IPv4 address and the TARGET parameter is A’s IPv4 address.

4. **Create a route for the tunnel**

   Add a route on each router, pointing to the IPv6 network on the other router.

   On router A, use the command:
   ```
   add ipv6 route=4ffe:1::/64 int=virt0
   ```

   On router B, use the command:
   ```
   add ipv6 route=3ffe:1::/64 int=virt0
   ```

   Note that the NEXTHOP parameter is not necessary for a tunnel.
5. **Test the routes.**

   On router A, use the command:
   
   `ping fe80::c0a8:0102%virt0`

   On router B, use the command:
   
   `ping fe80::c0a8:0101%virt0`

**IPv6 Filters**

This example illustrates how to configure IPv6 filters on the router. There are two hosts and Host A needs to be accessed by Telnet and FTP from Host B. Other traffic from B to A is blocked by the filter, except neighbour discovery, which is required for basic IPv6 routing. This configuration is shown in the following figure.

Figure 15-14: Example showing host and router connections

3. **To configure the required IPv6 filters**

   1. **Enable IPv6**

      Enable IPv6 by using the command:
      
      `enable ipv6`

   2. **Create the interfaces and assign IPv6 addresses to them, using the commands:**

      `CREATE ipv6 int=eth0`
      `CREATE ipv6 int=eth1`
      `add ipv6 int=eth0 ip=4ffe::0001/64`
      `add ipv6 int=eth1 ip=3ffe::0001/64`
3. Create the filter and add entries to it to allow Telnet and FTP traffic

To create the filter and add filter entries to enable Telnet and FTP access to Host A from Host B, use the commands:

```
add ipv6 filter=1 entry=1 source=3ffe::0005
destination=4ffe::0005 protocol=tcp sport=any dport=23
log=header
add ipv6 filter=1 entry=2 source=3ffe::0005
destination=4ffe::0005 protocol=tcp sport=any dport=21
log=header session=any
add ipv6 filter=1 entry=3 source=4ffe::0005
destination=3ffe::0005 session=any protocol=tcp
sport=any dport=ftpdata log=header
```

4. Add entries to the filter to allow Neighbour Discovery traffic

In addition to the Telnet and FTP filter entries, it is necessary to explicitly allow IPv6 Neighbour Discovery (ND) traffic from Host B to Host A, because this traffic is required to enable basic connectivity between devices. The following filter entries allow the types of ICMP traffic that are typically IPv6 ND traffic to pass through the router:

```
add ipv6 filter=1 entry=4 source=3ffe::/64 protocol=icmp
icmptype=135 icmpcode=any
add ipv6 filter=1 entry=5 source=3ffe::/64 protocol=icmp
icmptype=136 icmpcode=any
```

5. Assign the filter to the router’s interface to Host B

```
set ipv6 int=eth1 ipv6=3ffe::0001/64 filter=1
```

6. Check the configuration

The definitions of the filters on the router can be checked with the command:

```
show ipv6 filter
```

The following command displays details of the IP interfaces defined:

```
show ipv6 interface
```
add ipv6 6to4

Syntax
ADD IPV6 6TO4 IP=ipv4add

where ipv4add is a valid IPv4 address in dotted decimal notation

Description
This command configures the router to recognise 6-to-4 addresses and allows the router to act as a 6-to-4 relay. The router then performs automatic 6-to-4 tunnelling with the correct route configuration.

The IP parameter specifies the IPv4 address of an IPv4 interface on the router.

Example
To create a 6-to-4 tunnel, for traffic over an interface with the IPv4 address 202.101.202.101, use the command:

add ipv6 6to4 ip=202.101.202.101

Related Commands
add ipv6 tunnel
delete ipv6 6to4
show ipv6 tunnel
add ipv6 filter

Syntax

```
ADD IPV6 FILTER=filter-id SOURCE=ipv6addr/prefix-length
  [ACtion={INCLude|EXCLude}|PRIority=P0..P7]
  [DESTination=ipv6addr/prefix-length]  [DPort={port-name|
  port-id|ANY}]
  [ENTRY=entry-number]
  [ICMPCode={icmp-code-name|icmp-code-id|ANY}]
  [ICMPtrypE={icmp-type-name|icmp-type-id|ANY}]
  [LOG={4..1950|Dump|Header|None}]  [OPTIONS={YES|NO|ON|
  OFF|ENABLE|DISABLE}]
  [PROTocol={protocol|Any|Egp|Icmp|Ospf|Tcpp|Udp}]
  [SESSION={Any|Established|Start}]
  [SIZE={size|ANY}]
  [SPort={port-name|port-id|ANY}]
```

where:

- `filter-id` is a number from 0 to 99 or 200 to 299.
- `ipv6addr` is a valid IPv6 address, with its prefix length indicated by slash notation (see “IPv6 Interfaces and Addresses” on page 15-13).
- `prefix-length` is an integer from 1 to 128.
- `port-name` is the predefined name for a TCP or UDP port (see Table 15-3 on page 15-36).
- `port-id` is an IP port number or a range of port numbers in the format `[low]:[high]`.
- `entry-number` is the position of this entry in the filter.
- `icmp-code-name` is the predefined name for an ICMP reason code (see Table 15-4 on page 15-37).
- `icmp-code-id` is the number of an ICMP reason code (see Table 15-4 on page 15-37).
- `icmp-type-name` is the predefined name for an ICMP reason type (see Table 15-5 on page 15-37).
- `icmp-type-id` is the number of an ICMP reason type (see Table 15-5 on page 15-37).
- `protocol` is an IPv6 protocol number.
- `size` is a number from 0 to 65535.

Description

This command adds an entry to an IPv6 traffic filter or priority filter. The exact pattern within that entry should not already exist in the filter.

The FILTER parameter specifies the number of the filter to which the entry is to be added. Filters with numbers from 0 to 99 are treated as traffic filters, and use the ACTION parameter to specify the action to take with a packet that matches the entry. Filters with numbers from 200 to 299 are treated as priority filters, and use the PRIORITY parameter to specify the priority to assign to a packet that matches the entry. An interface may have a maximum of one traffic filter and one priority filter, but the same traffic or priority filter can be assigned to more than one interface. Traffic filters are applied to packets received via the interface, whereas priority filters are applied to packets as they are transmitted.

The SOURCE parameter specifies the source IP address, in IPv6 notation, for the entry. A prefix length can be specified using slash notation (e.g. 3FFE::0/16). The prefix is used to determine the portion of the source IPv6 address in...
the IPv6 packet that is significant for comparison with this entry. The default prefix length is 128.

The ACTION parameter specifies, for traffic filters, the action to take when the entry is matched. If INCLUDE is specified, the IP packet is processed and forwarded. If EXCLUDE is specified, the IP packet is discarded. The ACTION and PRIORITY parameters are mutually exclusive—only one may be specified. The default is INCLUDE.

The DESTINATION parameter specifies the destination IP address, in IPv6 notation, for the entry. A prefix length can be specified using slash notation (e.g. 3FFE::0/16). The prefix is used to determine the portion of the destination IPv6 address in the IPv6 packet that is significant for comparison with this entry. The default prefix length is 128.

The DPORT parameter specifies the port to check against the destination port for this entry, as the recognised name of a well-known UDP or TCP port, (see Table 15-3 on page 15-36), a decimal value from 0 to 65535, or a range of numbers formatted as low:high. If low is omitted, 0 is assumed. If high is omitted, the maximum port number is assumed. If a port other than ANY is specified, the PROTOCOL parameter must also be specified, and must be one of TCP or UDP. The default is ANY.

<table>
<thead>
<tr>
<th>Name</th>
<th>Port</th>
<th>Protocol</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ANY</td>
<td>-</td>
<td>-</td>
<td>Any port</td>
</tr>
<tr>
<td>BOOTPC</td>
<td>68</td>
<td>UDP</td>
<td>Bootstrap Protocol Client</td>
</tr>
<tr>
<td>BOOTPS</td>
<td>67</td>
<td>UDP</td>
<td>Bootstrap Protocol Server</td>
</tr>
<tr>
<td>DOMAIN</td>
<td>53</td>
<td>TCP/UDP</td>
<td>Domain Name Server</td>
</tr>
<tr>
<td>FINGER</td>
<td>79</td>
<td>TCP</td>
<td>Finger</td>
</tr>
<tr>
<td>FTP</td>
<td>21</td>
<td>TCP</td>
<td>File Transfer [Control]</td>
</tr>
<tr>
<td>FTPDATA</td>
<td>20</td>
<td>TCP</td>
<td>File Transfer [Default Data]</td>
</tr>
<tr>
<td>Gopher</td>
<td>70</td>
<td>TCP</td>
<td>Gopher</td>
</tr>
<tr>
<td>HOSTNAME</td>
<td>101</td>
<td>TCP/UDP</td>
<td>NIC Host Name Server</td>
</tr>
<tr>
<td>IPX</td>
<td>213</td>
<td>TCP/UDP</td>
<td>IPX</td>
</tr>
<tr>
<td>KERBEROS</td>
<td>88</td>
<td>UDP</td>
<td>Kerberos</td>
</tr>
<tr>
<td>LOGIN</td>
<td>49</td>
<td>UDP</td>
<td>Login Host Protocol</td>
</tr>
<tr>
<td>MSGICP</td>
<td>29</td>
<td>TCP/UDP</td>
<td>MSG ICP</td>
</tr>
<tr>
<td>NAMESERVER</td>
<td>42</td>
<td>UDP</td>
<td>Host Name Server</td>
</tr>
<tr>
<td>NEWS</td>
<td>144</td>
<td>TCP</td>
<td>NewS</td>
</tr>
<tr>
<td>NNTP</td>
<td>119</td>
<td>TCP</td>
<td>Network News Transfer Protocol</td>
</tr>
<tr>
<td>NTP</td>
<td>123</td>
<td>TCP</td>
<td>Network Time Protocol</td>
</tr>
<tr>
<td>RTELNET</td>
<td>107</td>
<td>TCP/UDP</td>
<td>Remote Telnet Service</td>
</tr>
<tr>
<td>SFTP</td>
<td>115</td>
<td>TCP/UDP</td>
<td>Simple File Transfer Protocol</td>
</tr>
<tr>
<td>SMTP</td>
<td>25</td>
<td>TCP</td>
<td>Simple Mail Transfer</td>
</tr>
<tr>
<td>SNMP</td>
<td>161</td>
<td>UDP</td>
<td>SNMP</td>
</tr>
<tr>
<td>SNMPTRAP</td>
<td>162</td>
<td>UDP</td>
<td>SNMPTRAP</td>
</tr>
<tr>
<td>SYSTAT</td>
<td>11</td>
<td>TCP</td>
<td>Active Users</td>
</tr>
</tbody>
</table>
The ENTRY parameter specifies the entry number that this new entry occupies in the filter. Existing entries with the same or higher entry numbers are pushed down the filter. The default is to add the new entry to the end of the filter.

The ICMPTYPE and ICMP_CODE parameters specify the ICMP message type and ICMP message reason code (see Table 15-4 on page 15-37) to match against the ICMP type and code fields in an ICMP packet. The ICMPTYPE parameter specifies the ICMP message type to match as a decimal value from 0 to 255, or the recognised name of an ICMP type (see Table 15-5 on page 15-37). A packet is matched against type and code when the PROTOCOL parameter is set to ICMP. The default is ANY.

Table 15-3: Well-known TCP/UDP ports (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Port</th>
<th>Protocol</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>TELNET</td>
<td>23</td>
<td>TCP</td>
<td>Telnet</td>
</tr>
<tr>
<td>TFTP</td>
<td>69</td>
<td>UDP</td>
<td>Trivial File Transfer</td>
</tr>
<tr>
<td>TIME</td>
<td>37</td>
<td>TCP/UDP</td>
<td>Time</td>
</tr>
<tr>
<td>UUCP</td>
<td>540</td>
<td>TCP</td>
<td>uucpd</td>
</tr>
<tr>
<td>UUCPRLOGIN</td>
<td>541</td>
<td>TCP/UDP</td>
<td>uucp-rlogin</td>
</tr>
<tr>
<td>XNSTIME</td>
<td>52</td>
<td>TCP/UDP</td>
<td>XNS Time Protocol</td>
</tr>
</tbody>
</table>

Table 15-4: Predefined ICMP code names used by the IPv6 filtering process

<table>
<thead>
<tr>
<th>ICMP code name</th>
<th>ICMP code value</th>
<th>Applies to ICMP type value</th>
<th>Command Line Syntax</th>
</tr>
</thead>
<tbody>
<tr>
<td>ICMPv6_ANY</td>
<td>-</td>
<td>-</td>
<td>ANY</td>
</tr>
<tr>
<td>ICMPv6_NO_ROUTE_TO_DESTINATION</td>
<td>0</td>
<td>1</td>
<td>NOROUTETODEST</td>
</tr>
<tr>
<td>ICMPv6_COMMUNICATION_PROHIBITED</td>
<td>1</td>
<td>1</td>
<td>COMMSPROHIBITED</td>
</tr>
<tr>
<td>ICMPv6_SCOPE_MISMATCH</td>
<td>2</td>
<td>1</td>
<td>SCOPEMISMATCH</td>
</tr>
<tr>
<td>ICMPv6_ADDRESS_UNREACHABLE</td>
<td>3</td>
<td>1</td>
<td>ADDRUNREACHABLE</td>
</tr>
<tr>
<td>ICMPv6_PORT_UNREACHABLE</td>
<td>4</td>
<td>1</td>
<td>PORTUNREACHABLE</td>
</tr>
<tr>
<td>ICMPv6_HOP_LIMIT_EXCEEDED</td>
<td>0</td>
<td>3</td>
<td>HOPLIMITEXCD</td>
</tr>
<tr>
<td>ICMPv6_REASSEMBLY_TIME_EXCEEDED</td>
<td>1</td>
<td>3</td>
<td>REASMBTIMEEXC</td>
</tr>
<tr>
<td>ICMPv6_ERRONEOUS_HEADER_FIELD</td>
<td>0</td>
<td>4</td>
<td>ERRONEOUSHEADER</td>
</tr>
<tr>
<td>ICMPv6_UNRECOGNISED_NEXT_HEADER</td>
<td>1</td>
<td>4</td>
<td>URCNXHEADER</td>
</tr>
<tr>
<td>ICMPv6_UNRECOGNISED_OPTION</td>
<td>2</td>
<td>4</td>
<td>URCOPTION</td>
</tr>
</tbody>
</table>

Table 15-5: Predefined ICMP type names used by the IPv6 filtering process

<table>
<thead>
<tr>
<th>ICMP Type Name</th>
<th>ICMP Type Value</th>
<th>ICMP Codes Supported</th>
<th>Command Line Syntax</th>
</tr>
</thead>
<tbody>
<tr>
<td>ICMPv6_ANY</td>
<td>-</td>
<td>-</td>
<td>ANY</td>
</tr>
<tr>
<td>ICMPv6_DESTINATION_UNREACHABLE</td>
<td>1</td>
<td>Yes</td>
<td>DESTUNREACH</td>
</tr>
<tr>
<td>ICMPv6_PACKET_TOO_BIG</td>
<td>2</td>
<td>Yes</td>
<td>PKTTOOBIG</td>
</tr>
<tr>
<td>ICMPv6_TIME_EXCEEDED</td>
<td>3</td>
<td>Yes</td>
<td>TIMEEXCEEDED</td>
</tr>
<tr>
<td>ICMPv6_PARAMETER_PROBLEM</td>
<td>4</td>
<td>Yes</td>
<td>PARAMPROB</td>
</tr>
</tbody>
</table>
The LOG parameter specifies whether matches to a filter entry result in a log message being sent to the router’s logging facility, and the content of the log messages. This parameter enables logging of the IP packet filtering process down to the level of an individual filter entry.

- If a number from 4 to 1950 is specified, the first 4 to 1950 octets of the data portion of TCP, UDP and ICMP packets or the first 4 to 1950 octets after the IP header of other protocol packets are logged with a message type/subtype of IPFIL/DUMP. The filter number, entry number and IP header information (source and destination IP addresses, protocol, source and destination ports, and size) are also logged with a message type/subtype of IPFIL/PASS (for entries with an INCLUDE action) or IPFIL/FAIL (for entries with an EXCLUDE action).

- If DUMP is specified, the filter number, entry number and IP header information (source and destination IP addresses, protocol, source and destination ports, and size) are logged with a message type/subtype of IPFIL/DUMP. In addition, the first 40 octets of the data portion of TCP, UDP and ICMP packets, or the first 40 octets after the IP header of other protocol packets are logged with a message type/subtype of IPFIL/DUMP.

- If HEADER is specified, the filter number, entry number and IP header information (source and destination IP addresses, protocol, source and destination ports, and size) are logged with a message type/subtype of IPFIL/PASS (for entries with an INCLUDE action) or IPFIL/FAIL (for entries with an EXCLUDE action).

- If NONE is specified, matches to the filter entry are not logged.

The OPTIONS parameter specifies the presence or absence of any time-length-variable (TLV) encoded “Options” to check against for this entry. TLV-encoded options can be found in the Hop-by-Hop and Destination Options extension headers. If YES is specified, the entry matches IP packets with any extension header TLV options set. If NO is specified, the entry matches IP packets without any extension header TLV options set. The default is NO.

<table>
<thead>
<tr>
<th>ICMP Type Name</th>
<th>ICMP Type Value</th>
<th>ICMP Codes Supported</th>
<th>Command Line Syntax</th>
</tr>
</thead>
<tbody>
<tr>
<td>ICMPv6_ECHO_REQUEST</td>
<td>128</td>
<td>No</td>
<td>ECHORQ</td>
</tr>
<tr>
<td>ICMPv6_ECHO_REPLY</td>
<td>129</td>
<td>No</td>
<td>ECHORP</td>
</tr>
<tr>
<td>ICMPv6_MULTICAST_LISTENER_QUERY</td>
<td>130</td>
<td>No</td>
<td>MLQUERY</td>
</tr>
<tr>
<td>ICMPv6_MULTICAST_LISTENER_REPORT</td>
<td>131</td>
<td>No</td>
<td>MLREP</td>
</tr>
<tr>
<td>ICMPv6_MULTICAST_LISTENER_DONE</td>
<td>132</td>
<td>No</td>
<td>MLDONE</td>
</tr>
<tr>
<td>ICMPv6_ROUTER_SOLICIT</td>
<td>133</td>
<td>No</td>
<td>RTSOLICIT</td>
</tr>
<tr>
<td>ICMPv6_ROUTER_ADVERT</td>
<td>134</td>
<td>No</td>
<td>RTADVERT</td>
</tr>
<tr>
<td>ICMPv6_NEIGHBOR_SOLICIT</td>
<td>135</td>
<td>No</td>
<td>NBRSOCLIT</td>
</tr>
<tr>
<td>ICMPv6_NEIGHBOR_ADVERT</td>
<td>136</td>
<td>No</td>
<td>NBRADVERT</td>
</tr>
<tr>
<td>ICMPv6_REDIRECT</td>
<td>137</td>
<td>No</td>
<td>REDIRECT</td>
</tr>
<tr>
<td>ICMPv6_ROUTER_RENUMBERING</td>
<td>138</td>
<td>No</td>
<td>RTRENUMBER</td>
</tr>
</tbody>
</table>

Table 15-5: Predefined ICMP type names used by the IPv6 filtering process (continued)
The PRIORITY parameter specifies, for priority filters, the priority to apply to forwarding packets when the entry is matched. A low value (P0) assigns a high priority to the packet. A high value (P7) assigns a low priority to the packet. The priority number is employed during forwarding (transmission). The default is P7. The ACTION and PRIORITY parameters are mutually exclusive—only one may be specified.

The PROTOCOL parameter specifies a protocol to check against the protocol for this entry, as a decimal value from 0 to 65534, or the recognised name of an IP protocol type. If either the SPORT or DPORT parameters are used, PROTOCOL must be defined as TCP, UDP or ANY. Specifying TCP or UDP filters packets from companion protocols, such as ICMP and OSPF, that do not use TCP or UDP as a transport mechanism. The default is ANY.

The SESSION parameter specifies the type of TCP packet to match and is used as a basis for packet filtering when the PROTOCOL parameter specifies TCP. If START is specified, the entry matches TCP packets with the SYN bit set and the ACK bit clear. If ESTABLISHED is specified, the entry matches TCP packets with either the SYN bit clear or the ACK bit set. If ANY is specified, the entry matches any TCP packet. The default is ANY.

The SIZE parameter specifies the maximum unassembled size to match against, for each IP packet or fragment. If the fragment’s offset plus size is greater than the value specified, the fragment is discarded. The default is ANY, which indicates that size is not required as a matching category.

The SPORT parameter specifies the port to check against the source port for this entry, as the recognised name of a well-known UDP or TCP port, (see Table 15-3 on page 15-36), a decimal value from 0 to 65535, or a range of numbers in the form [low]:[high]. If low is omitted, 0 is assumed. If high is omitted, the maximum port number is assumed. If a port other than ANY is specified, the PROTOCOL parameter must also be specified, and must be one of TCP or UDP. The default is ANY.

Examples

To add a filter to block all Telnet traffic, from IP address 3FFE::3 to any IP address whose first 64 contiguous bits match 3FFE::4, and log the header details, use the command:

```
add ipv6 fi=2 so=3ffe::3/128 des=3ffe::4/64 si=ANY prot=tcp ac=excl sp=any dp=telnet log=h
```

Related Commands

- `add ipv6 interface`
- `delete ipv6 filter`
- `set ipv6 filter`
add ipv6 host

Syntax  

ADD IPV6 HOST=name IPaddress=ipv6add [INTERFACE=interface]

where:

- **name** is a character string 1 to 60 characters long. Valid characters are any printable characters. If the string contains spaces it must be in double quotes.
- **ipv6add** is a valid IPv6 address.
- **interface** is a valid interface.

Description  

This command adds a user-defined name for an IPv6 host to the router’s host name table. The host name table makes it easier to Telnet to commonly accessed hosts, by allowing the user to enter a shorter, easier to remember name for the host rather than the host’s full IPv6 address or domain name.

The HOST parameter specifies the user-defined name for the IPv6 host. A host with the same name must not already exist in the host name table. When a host name is specified in the **telnet** command (see the **telnet** command on page 21-31 of Chapter 21, Terminal Server), the entire name is used to match a name in the host name table. All characters are used in the comparison, including non-alphabetic characters when they are present. The HOST parameter is **not** case-sensitive so the names *MyName* and *myname* are identical.

The IPADDRESS parameter specifies the IPv6 address of the host.

The INTERFACE parameter specifies the interface, on the router from which the Telnet request originates, that is then connected to the required destination interface. Telneting to a link-local address requires interface information as well as the address because a single link-local address can belong to several interfaces. The INTERFACE parameter is mandatory only if a user is adding a host entry that uses a link-local address. Valid interfaces are:

- eth (e.g. eth0)
- PPP (e.g. ppp0)
- VLAN (e.g. vlan1)
- virtual tunnel (e.g. virt9)

To see a list of current valid interfaces, use the commands **show interface** command on page 7-66 of Chapter 7, Interfaces.

Examples  

To add a new host called *foobar* with an IPv6 address of 3FFE::1, use the command:

```
add ipv6 ho=foobar ip=3ffe::1
```

Related Commands  

**delete ipv6 host**  
**show ipv6 host**
add ipv6 interface

**Syntax**

ADD IPV6 INTERFACE=interface
IPaddress=(ipv6add/prefix-length|DHCP|DHCPTEMP|PD) [APPInterface=app-interface[,...]] [DECrement={YES|NO|ON|OFF|ENAble|DISable}] [FILter=0..99] [HInt=ipv6add/prefix-length] [KEY=key-id] [PREFerred=1..4294967295|INFinite] [PRIorityfilter=200..299] [PUBLish={YES|NO|ON|OFF|ENAble|DISable}] [TYPe={ANYcast|UNIcast}] [VALid=1..4294967295|INFinite]

where:

- **interface** is a valid interface.
- **ipv6add** is a valid IPv6 address, with its prefix length indicated by slash notation (see “IPv6 Interfaces and Addresses” on page 15-13).
- **prefix-length** is an integer from 1 to 128.
- **app-interface** is a valid interface, optionally followed by a slash and an integer from 1 to 65535 (e.g. ppp1/2).
- **key-id** is a number from 0 to 65535.

**Description**

This command adds an IPv6 address to an interface, with a specified lifetime. If no lifetime is specified then the lifetime is infinite. If the interface has not already been created, this command creates it.

The INTERFACE parameter is the interface to add the IPv6 address to. Valid interfaces are:

- eth (e.g. eth0)
- PPP (e.g. ppp0)
- VLAN (e.g. vlan1)
- virtual tunnel (e.g. virt9)

To see a list of current valid interfaces, use the commands **show interface** command on page 7-66 of Chapter 7, Interfaces.

The IPADDRESS parameter specifies one of:

- the IPv6 address to assign to the interface, in IPv6 notation. A prefix length must be specified, using slash notation (e.g. 3FFE::0/16)
- that the interface request a permanent IPv6 address from a DHCPv6 server (the DHCP option)
- that the interface request a temporary IPv6 address from a DHCPv6 server (the DHCPv6 option). Using temporary addresses increases security
- that the interface request a prefix from a DHCPv6 server in order for the router to apply the prefix to other IPv6 interfaces (the PD option). The APPINT parameter must also be specified and lists interfaces to which the prefix is assigned.

An IPv6 interface that is used to as a DHCP server must not use DHCP to resolve its IPv6 address.
The APPINT parameter is required when the interface obtains a prefix by prefix delegation from DHCP6 in order to assign the prefix to other interfaces. It specifies the interface or comma-separated interfaces to which the received prefix is assigned. APPINT is required if the IPADDRESS parameter is set to PD. Valid interfaces are:

- eth (e.g. eth0)
- PPP (e.g. ppp0)
- VLAN (e.g. vlan1)
- virtual tunnel (e.g. virt9)

To see a list of current valid interfaces, use the commands *show interface command on page 7-66 of Chapter 7, Interfaces*. The interface is optionally followed by a value that determines the next 16 bits for the prefix assigned to this interface, if the delegated prefix has a prefix length of less than 64 bits (for example, ppp1/2). If a value is not specified, the router assigns a value of 1 to the first APPINT, 2 to the second APPINT, and so on.

The DECREMENT parameter specifies whether the time counters for valid and preferred IPv6 address lifetimes, and published prefix lifetimes are decremented. IPv6 addresses with valid and preferred lifetimes set to infinite are not decremented. The default is NO.

The FILTER parameter is the number of the traffic filter that is to be applied to this interface.

The HINT parameter is used with DHCP6 requests. It specifies a particular IPv6 address to request, if IPADDRESS is DHCP or DHCPTEMP, or a particular prefix and/or prefix length to request if IPADDRESS is PD. Note that requesting a particular address or prefix is no guarantee of receiving it because the server may already have assigned it to another client. To guarantee that a client always receives a particular address or prefix, create a static entry by using the command *add dhcp6 range in Chapter 36, Dynamic Host Configuration Protocol for IPv6 (DHCP6)*.

The KEY parameter is used to configure authentication of DHCP6 message exchanges, and specifies the ID number of the authentication key.

The PREFERRED parameter specifies the time in seconds for which this IPv6 address is to be the preferred address for this interface. The default is 604800 seconds (7 days). The value of this parameter cannot be greater than that of the VALID parameter.

The PRIORITYFILTER parameter is the number of the priority filter that is to be attached to this interface.

The PUBLISH parameter determines whether to include the prefix in router advertisement packets. The default is NO. Note that, to comply with Section 6.2.1 of RFC 2461, by default the router does not generate router advertisements. To enable the router to generate router advertisements that include this router’s address, you must specify PUBLISH=YES, and also enable advertisements using the *enable ipv6 advertise command on page 15-58*. If the router does not generate router advertisements, the router discovery and address autoconfiguration processes in the network may not work as expected.

The TYPE parameter allows an anycast address to be distinguished from a unicast address. The default is UNICAST.
The VALID parameter is the time in seconds that the IPv6 address exists on the interface. After this time, the IPv6 address is deleted. The default is 2592000 seconds (30 days). The time value must be the same as or greater than that of the PREFERED parameter.

**Examples**

To add an IPv6 address of 3FFE::1/32 to eth0, use the command:

```
add ipv6 ip=3ffe::/32 int=eth0
```

To request an IPv6 prefix with a prefix length of 48 bits from eth0 to be delegated on eth1, use the command:

```
add ipv6 int=eth0 ip=pd appint=eth1 hi=::/48
```

**Related Commands**

- create ipv6 interface
- delete ipv6 interface
- destroy ipv6 interface
- set ipv6 interface
- show ipv6
- show ipv6 interface

### add ipv6 nd

**Syntax**

```
ADD IPV6 ND=ipv6add INTERFACE=interface ETHernet=macadd
   [ISRouter={YES|NO}] [POrt=port-number]
```

where:

- **ipv6add** is a valid IPv6 address.
- **interface** is a valid interface.
- **macadd** is a MAC address in hexadecimal notation.
- **port-number** is the physical switch port number. Port numbers start at 1 and end at $m$, where $m$ is the highest numbered Ethernet switch port, including plunk ports.

**Description**

This command manually adds a neighbour to the neighbour cache.

The ND parameter specifies the IPv6 address of the neighbour.

The INTERFACE parameter specifies the interface on which the router receives data from the neighbour. Valid interfaces are:

- **eth** (e.g. eth0)
- **PPP** (e.g. ppp0)
- **VLAN** (e.g. vlan1)
- **virtual tunnel** (e.g. virt9)
To see a list of current valid interfaces, use the commands `show interface` command on page 7-66 of Chapter 7, Interfaces.

The ETHERNET parameter specifies the MAC address (the Ethernet link-layer address) of the neighbour.

The ISROUTER parameter specifies whether the added neighbour is a router. The default is NO.

The PORT parameter specifies the physical switch port number in a VLAN. If the INTERFACE parameter specifies a VLAN interface, the PORT parameter is required. Otherwise the PORT parameter is not valid.

**Examples**

To manually add a host as a neighbour, with the MAC address aa-bb-cc-dd-11-22 and the IPv6 address 3ffe::1, to the eth0 interface, use the command:

```
add ipv6 nd=3ffe::1 int=eth0 eth=aa-bb-cc-dd-11-22
```

**Related Commands**

- `delete ipv6 nd`
- `reset ipv6 ndcache`
- `show ipv6 ndcache`

### add ipv6 prefix

**Syntax**

```
ADD IPV6 PREFIX=ipv6add/prefix-length INTERFACE=interface
[AUTOnomous={YES|NO}] [ONlink={YES|NO}]
[PREFerred=1..4294967295|INFinite]
[VALid=1..4294967295|INFinite]
```

where:

- `ipv6add` is a valid IPv6 prefix, with the prefix length indicated by slash notation (see “IPv6 Interfaces and Addresses” on page 15-13).
- `prefix-length` is the number between 1 and 128.
- `interface` is a valid interface.

**Description**

This command specifies a prefix for the router to include in its advertisement messages for the specified interface, without adding the prefix to the interface’s IPv6 address. The prefix can assist nodes on the subnet with configuration of their local-link IPv6 addresses. More than one prefix can be included in the router advertisements for a given interface.

If the PUBLISH parameter is set to YES for an interface using the `add ipv6 interface` command on page 15-41 or the `set ipv6 interface` command on page 15-64, the prefix specified by that command is advertised as well as the prefix specified here.

The PREFIX parameter specifies the IPv6 prefix (e.g. 3ffe::/64) to advertise for this interface.

The INTERFACE parameter is the interface to add the IPv6 address to. Valid interfaces are:
To see a list of current valid interfaces, use the commands `show interface` command on page 7-66 of Chapter 7, Interfaces.

The AUTONOMOUS parameter specifies whether the prefix is for the same autonomous system (for example, the Internet or an intranet). If it is, hosts use this prefix to configure their addresses over this autonomous system. The default is YES.

The ONLINK parameter specifies whether hosts should use this prefix to configure their addresses on the local link. The default is YES.

The PREFERRED parameter specifies the time in seconds for which the prefix is preferred over other prefixes. The prefix no longer appears in router advertisements after this period. The default is 604800 seconds (7 days).

The VALID parameter specifies the time in seconds for which the prefix is valid. The prefix is deleted after this period. The default is 2592000 seconds (30 days).

**Examples**
To include the prefix 3ffe::/64 in router advertisements sent over the eth0 interface, use the command:

```
add ipv6 prefix=3ffe::/64 int=eth0
```

**Related Commands**
- delete ipv6 prefix
- set ipv6 prefix
- show ipv6 ndcache

---

**add ipv6 rip**

**Syntax**
```
ADD IPV6 RIP INTERFACE=interface POISONREVERSE=(ON|OFF|True|False)
```

**Description**
This command enables the router to listen for RIPv6 packets on the specified interface.

The INTERFACE parameter is the physical interface to listen for RIP packets on. Valid interfaces are:

- eth (e.g. eth0)
- PPP (e.g. ppp0)
- VLAN (e.g. vlan1)
- virtual tunnel (e.g. virt9)
To see a list of current valid interfaces, use the commands `show interface` command on page 7-66 of Chapter 7, Interfaces.

The POISONREVERSE parameter specifies whether poison reverse is enabled on the interface. To enable poison reverse, specify ON or TRUE. To disable poison reverse, specify OFF or FALSE. Poison reverse addresses the problem of slow convergence on RIPng routes. If one device in a network goes down, it can take a long time for the devices to recognise that routes through the crashed device are no longer available. If poison reverse is enabled when a device goes down, the devices next to it continues to advertise the route but with a cost of 16. This cost indicates that the route is unavailable.

**Examples**

To enable listening for RIP packets on eth0, use the command:

```
add ipv6 rip int=eth0 poi=on
```

**Related Commands**
- delete ipv6 rip
- disable ipv6 rip
- enable ipv6 rip
- show ipv6 rip

### add ipv6 route

**Syntax**

```
ADD IPV6 ROUte=ipv6add/prefix-length INTERFACE=interface
[NEXThop=ipv6add] [METric=1..16] [PREFerence=0..65535]
```

where:
- `prefix-length` is an integer from 1 to 128.
- `ipv6add` is a valid IPv6 address, with its prefix length indicated by slash notation (see “IPv6 Interfaces and Addresses” on page 15-13).
- `interface` is a valid interface.

**Description**

This command adds a static route to the IPv6 route table. The static route must not already exist. However, if the route exists as a dynamic (e.g. RIP-derived) route, the static route may still be added. To specify that the destination node is on the network to which traffic is being directed, the NEXTHOP parameter should be `0:0:0:0:0:0:0:0` (or `::`).

Static routes can be used to define default routes to external routers or networks. A default route is one with a network address of `0:0:0:0:0:0:0:0` (or `::`). If the router receives data and cannot find a route for it, the data is sent to the default route. To define a default route, set ROUTE to `::` and set NEXTHOP to point to the network (router) to which default packets are to be directed.

The ROUTE parameter is the IPv6 address of a host or network to which packets are to be routed. Slash notation can be used on this parameter.

The INTERFACE parameter is the physical interface over which the router forwards packets. Valid interfaces are:
- `eth` (e.g. `eth0`)
- `PPP` (e.g. `ppp0`)
- `VLAN` (e.g. `vlan1`)
- `virtual tunnel` (e.g. `virt9`
To see a list of current valid interfaces, use the commands show interface command on page 7-66 of Chapter 7, Interfaces.

The NEXTHOP parameter is the IPv6 address of the next router along the path to the destination, and is required for all interfaces except static tunnel virtual (VIRT) interfaces and PPP. For routes through 6-to-4 tunnel VIRT interfaces, it is required and is the IPv6 address of the router at the other end of the tunnel.

The METRIC parameter specifies the cost of the route. The cost is used in RIP entries to determine the best path to a node. The default is 1.

The PREFERENCE parameter specifies the preference for the route. When more than one route in the route table matches the destination address in an IPv6 packet, the route with the lowest preference value is used to route the packet. If two or more candidate routes have the same preference, the route with the longest prefix is used. Interface routes have a preference of 0 and RIP routes have a preference of 100. The default preference for static routes other than 0:0:0:0:0:0:0 is 60. The default for the default static route 0:0:0:0:0:0:0:0 is 360.

**Examples**  To add a route to the network 3FFE::/16, using a gateway with the IPv6 address 4FFE::1 that is connected to the router on eth0, use the command:

```
add ipv6 ro=3ffe::/16 next=4ffe::1 int=eth0
```

**Related Commands**  
- delete ipv6 route
- show ipv6
- show ipv6 route
add ipv6 tunnel

Syntax
ADD IPV6 TUNnel LOcal=ipv4add TArget=ipv4add
[INTerface=tunnel-interface] [IPaddress=ipv6add]

where:

- ipv4add is a valid IPv4 address in dotted decimal notation.
- tunnel-interface is a virtual interface of the form virtn, where n is an integer between 0 and 255.
- ipv6add is a valid IPv6 address (see “IPv6 Interfaces and Addresses” on page 15-13).

Description
This command creates a static tunnel between two IPv6 networks, to link them over an IPv4 network. A tunnel must be created on both of the IPv6 switches or routers that link the two networks, and a route that uses the tunnel must be added on each end.

The LOCAL parameter is the IPv4 address of the interface on the router that is used in the tunnel.

The TARGET parameter is the IPv4 address of the last node in the tunnel. The end node must be capable of forwarding an IPv6 packet. A tunnel must also be configured on the remote router or switch.

The INTERFACE parameter specifies a virtual tunnel interface for this tunnel (for example, virt5). If this parameter is not specified, the tunnel is created on the first available virtual interface.

The IPADDRESS parameter specifies the IPv6 address of the tunnel, if an address is being added to the tunnel manually. If this parameter is not specified, the router assigns a link-local IPv6 address to the tunnel (see “Static Tunnelling” on page 15-22).

Examples
To add a tunnel between two routers A and B, use the following commands on router A:

```
add ip int=eth0 ip=192.168.1.1
add ipv6 tun lo=192.168.1.1 ta=192.168.1.2
```

Add the following commands on router B:

```
add ip int=eth0 ip=192.168.1.2
add ipv6 tun lo=192.168.1.2 ta=192.168.1.1
```

Related Commands
add ipv6 route
delete ipv6 tunnel
show ipv6 tunnel
create ipv6 interface

Syntax
CREate IPV6 INTerface=interface [DUPtrans=1..16]
[RETRans=0..4294967295]

where interface is a valid interface

Description
This command creates an IPv6 Ethernet interface and uses stateless address
autoconfiguration to assign it a link-local address. The address is formed from
the link-layer address and the prefix FE80::.

To create a PPP interface and assign it an IPv6 address, use the add ipv6
interface command on page 15-41. To create a VIRT interface, use the add ipv6
tunnel command on page 15-48.

The INTERFACE parameter specifies the interface that is to be created for IPv6.
Routing using this interface is then possible. Valid interfaces are:
- eth (e.g. eth0)
- PPP (e.g. ppp0)
- VLAN (e.g. vlan1)
- virtual tunnel (e.g. virt9)

To see a list of current valid interfaces, use the commands show interface
command on page 7-66 of Chapter 7, Interfaces.

The DUPTRANS parameter is the DupAddrDetectTransmits value. This is the
number of Neighbour Solicitation messages that the router sends while
performing Duplicate Address Detection on a tentative address. The default is
1.

The RETRANS parameter is the AdvRetransTimer, in milliseconds. This is the
interval between repeats of each Router Advertisement message sent by the
router. The default is 0, which indicates that this timer is not specified by this
router. If a RETRANS below 100 is specified, other than 0, the value is set to 100
milliseconds. The default is 0, which indicates that this timer is not specified by
this router.

Examples
To create an IPv6 interface, use the command:
cre ipv6 int=eth0

Related Commands
add ipv6 interface
add ipv6 tunnel
delete ipv6 interface
destroy ipv6 interface
set ipv6 interface
set ipv6 nd
show ipv6
show ipv6 interface
delete ipv6 6to4

**Syntax**
DELete IPV6 6TO4 IP=ipv4add

where `ipv4add` is a valid IPv4 address in dotted decimal notation

**Description**
This command removes the specified 6-to-4 interface from the router.

The IP parameter specifies the IPv4 address of the 6-to-4 interface that is to be removed.

**Related Commands**
**add ipv6 6to4**
**add ipv6 tunnel**
**show ipv6 tunnel**

delete ipv6 filter

**Syntax**
DELete IPV6 FILTER=filter-id ENTRY=(entry-number|ALL)

where:
- `filter-id` is a number from 0 to 99 or 200 to 299.
- `entry-number` is the position of this entry in the filter.

**Description**
This command deletes an entry from an IP traffic filter or priority filter. The entry must already exist in the filter.

The FILTER parameter specifies the number of the filter from which the entry is to be deleted. Filters with numbers from 0 to 99 are traffic filters and filters with numbers from 200 to 299 are priority filters.

The ENTRY parameter specifies the entry number in the filter that is to be deleted. Existing entries with the same or higher entry numbers are pushed up the filter to occupy the vacant entry. If ALL is specified, the filter is deleted.

**Examples**
To delete entry 2 from filter 2, use the command:
```
del ipv6 fil=2 ent=2
```

To delete all entries from filter 2, use the command:
```
del ipv6 fil=2 ent=all
```

**Related Commands**
**add ipv6 filter**
**set ipv6 filter**
delete ipv6 host

Syntax
DELete IPV6 HOst=name

where name is a character string 1 to 60 characters long

Description
This command removes a host name from the host name table.

The HOST parameter is a name that has been associated with an IPv6 address by using the add ipv6 host command.

Examples
To remove the host entry foobar, use the command:
   del ipv6 ho=foobar

Related Commands
add ipv6 host
show ipv6 host

delete ipv6 interface

Syntax
DELete IPV6 INTerface=interface IPaddress=ipv6add

where:
■ interface is a valid interface.
■ ipv6add is a valid IPv6 address.

Description
This command removes an IPv6 address from an interface.

The INTERFACE parameter specifies the interface on the router to remove the IPv6 address from. The interface must already be assigned and configured. Valid interfaces are:
■ eth (e.g. eth0)
■ PPP (e.g. ppp0)
■ VLAN (e.g. vlan1)
■ virtual tunnel (e.g. virt9)

To see a list of current valid interfaces, use the show interface command on page 7-66 of Chapter 7, Interfaces, or the show ipv6 interface command on page 15-79.

The IPADDRESS parameter specifies the IPv6 address to remove from the interface.

Examples
To remove the address 3FFE::1 from the eth0 interface, use the command:
   del ipv6 int=eth0 ip=3ffe::1

Related Commands
add ipv6 interface
create ipv6 interface
destroy ipv6 interface
set ipv6 interface
show ipv6
delete ipv6 nd

Syntax

DELete IPV6 ND=ipv6add INTERFACE=interface

where:

- **ipv6add** is a valid IPv6 address.
- **interface** is a valid interface.

Description

This command manually deletes a neighbour from the neighbour cache.

The ND parameter specifies the IPv6 address of the neighbour.

The INTERFACE parameter specifies the interface on which the router receives data from the neighbour. The interface must already be assigned and configured. Valid interfaces are:

- eth (e.g. eth0)
- PPP (e.g. ppp0)
- VLAN (e.g. vlan1)
- virtual tunnel (e.g. virt9)

To see a list of current valid interfaces, use the show interface command on page 7-66 of Chapter 7, Interfaces, or the show ipv6 interface command on page 15-79.

Examples

To specify that the host with IPv6 address 3ffe::1 is no longer a neighbour on the eth0 interface, use the command:

```
del ipv6 nd=3ffe::1 int=eth0
```

Related Commands

- add ipv6 nd
- reset ipv6 ndcache
- show ipv6 ndcache
delete ipv6 prefix

**Syntax**
```
DELete IPV6 PREFIX=ipv6add/prefix-length
    INTERface=interface
```

where:

- `ipv6add` is a valid IPv6 prefix, with the prefix length indicated by slash notation (see "IPv6 Interfaces and Addresses" on page 15-13).
- `prefix-length` is the number between 1 and 128.
- `interface` is a valid interface.

**Description**
This command removes the IPv6 prefix from the prefix list. This prevents this prefix from being advertised via Router Advertisements.

The INTERFACE parameter specifies the interface on the router to remove the IPv6 prefix from. The interface must already be assigned and configured. Valid interfaces are:

- `eth` (e.g. eth0)
- `PPP` (e.g. ppp0)
- `VLAN` (e.g. vlan1)
- `virtual tunnel` (e.g. virt9)

To see a list of current valid interfaces, use the `show interface` command on page 7-66 of Chapter 7, Interfaces, or the `show ipv6 interface` command on page 15-79.

**Related Commands**
- `add ipv6 prefix`
- `set ipv6 prefix`
- `show ipv6 ndcache`

---

delete ipv6 rip

**Syntax**
```
DELete IPV6 RIP INTERface=interface
```

where `interface` is a valid interface

**Description**
This command stops an interface from sending or receiving RIP packets.

The INTERFACE parameter specifies the physical interface that the router was sending and receiving RIP packets on. The interface must already be assigned and configured. Valid interfaces are:

- `eth` (e.g. eth0)
- `PPP` (e.g. ppp0)
- `VLAN` (e.g. vlan1)
- `virtual tunnel` (e.g. virt9)
To see a list of current valid interfaces, use the `show interface` command on page 7-66 of Chapter 7, Interfaces, or the `show ipv6 interface` command on page 15-79.

**Examples**

To stop the eth0 interface from sending or receiving RIP packets, use the command:

```
   del ipv6 rip int=eth0
```

**Related Commands**

- `add ipv6 rip`
- `disable ipv6 rip`
- `enable ipv6 rip`
- `show ipv6 interface`

---

**delete ipv6 route**

**Syntax**

```
DELete IPV6 ROUTe=ipv6add INTERFACE=interface
   NEXThop=ipv6add
```

where:

- `ipv6add` is a valid IPv6 address.
- `interface` is a valid interface.

**Description**

This command removes a route from the IPv6 route table.

The ROUTE parameter specifies the IPv6 address of a host or network to which packets were being routed.

The INTERFACE parameter specifies the physical interface out of which the router was forwarding packets. The interface must already be assigned and configured. Valid interfaces are:

- eth (e.g. eth0)
- PPP (e.g. ppp0)
- VLAN (e.g. vlan1)
- virtual tunnel (e.g. virt9)

To see a list of current valid interfaces, use the `show interface` command on page 7-66 of Chapter 7, Interfaces, or the `show ipv6 interface` command on page 15-79.

The NEXTHOP parameter specifies the IPv6 address of the router that was next on the path to the destination.

**Examples**

To delete the route to 3FFE:: with a next hop of 3FFE::1 on eth0, use the command:

```
   del ipv6 rou=3ffe:: nest=3ffe::1 int=eth0
```

**Related Commands**

- `add ipv6 route`
- `show ipv6 route`
delete ipv6 tunnel

Syntax
DELete IPV6 TUNnel=ipv6add INTerface=tunnel-interface

where:
- `ipv6add` is the first link-local address on the tunnel interface (the address that the router assigned to the tunnel when it was created).
- `tunnel-interface` is a virtual interface of the form virtn, where n is an integer between 0 and 255.

Description
This command removes an IPv6 tunnel. The tunnel’s address and interface can be seen by using the `show ipv6 tunnel` command on page 15-91.

The TUNNEL parameter specifies the IPv6 address of the tunnel to be removed. The tunnel’s IPv6 address is the address that was assigned to the tunnel interface when the tunnel was created.

The INTERFACE parameter specifies the virtual interface from which the tunnel is to be removed.

Examples
To delete the tunnel with an IPv6 address of fe80::c0a8:0102 on virt1, use the command:
```
del ipv6 tun=fe80::c0a8:0102 int=virt1
```

Related Commands
- add ipv6 tunnel
- show ipv6 tunnel

---

destroy ipv6 interface

Syntax
DESTroy IPV6 INTerface=interface

where `interface` is a valid interface name

Description
This command removes an IPv6 interface.

The INTERFACE parameter specifies the physical interface that is to be removed from IPv6. Routing using this interface is no longer possible. The interface must already be assigned and configured. Valid interfaces are:
- eth (e.g. eth0)
- PPP (e.g. ppp0)
- VLAN (e.g. vlan1)
- virtual tunnel (e.g. virt9)

To see a list of current valid interfaces, use the `show interface` command on page 7-66 of Chapter 7, Interfaces, or the `show ipv6 interface` command on page 15-79.
Examples  To remove the eth0 interface, use the command:

```
dest ipv6 int=eth0
```

Related Commands  
- add ipv6 interface
- create ipv6 interface
- delete ipv6 interface
- set ipv6 interface
- show ipv6 interface

---

disable ipv6

Syntax  DISable IPV6

Description  This command disables the IPv6 module. Any dynamic configuration, such as address autoconfiguration, is preserved between disabling and re-enabling the IPv6 module. The IPv6 module is disabled by default.

Examples  To disable the IPv6 module, use the command:

```
dis ipv6
```

Related Commands  
- enable ipv6
- show ipv6

---

disable ipv6 advertise

Syntax  DISable IPV6 ADVertise [INTerface=interface]

where interface is a valid interface

Description  This command disables sending of router advertisement packets. Router advertisements are disabled by default.

If the INTERFACE parameter is specified, sending router advertisements is enabled on that interface. Otherwise, sending router advertisement packets is enabled for all IPv6 interfaces. The interface must already be assigned and configured. Valid interfaces are:

- eth (e.g. eth0)
- PPP (e.g. ppp0)
- VLAN (e.g. vlan1)
- virtual tunnel (e.g. virt9)

To see a list of current valid interfaces, use the show interface command on page 7-66 of Chapter 7, Interfaces, or the show ipv6 interface command on page 15-79.
Examples  To disable router advertisements on all interfaces, use the command:

```
dis ipv6 adv
```

Related Commands  enable ipv6 advertise
show ipv6

disable ipv6 debug

Syntax  DISable IPV6 DEBug

Description  This command disables debugging of IPv6 packets. Debugging is disabled by default.

Examples  To disable IPv6 debugging, use the command:

```
dis ipv6 deb
```

Related Commands  show ipv6
enable ipv6 debug

disable ipv6 mtudiscovery

Syntax  DISable IPV6 MTUdiscovery

Description  This command disables Path MTU Discovery. The router does not automatically increase the Path MTU and ignores all ICMPv6 Packet Too Big Messages. Path MTU Discovery is disabled by default.

Examples  To disable Path MTU Discovery, use the command:

```
dis ipv6 mtu
```

Related Commands  enable ipv6 mtudiscovery
disable ipv6 rip

**Syntax**

DISable IPV6 RIP

**Description**

This command disables routing of IPv6 packets using RIPv6. RIP is disabled by default.

**Examples**

To disable the sending and receiving of RIP packets, use the command:

```
dis ipv6 rip
```

**Related Commands**

- add ipv6 rip
- delete ipv6 rip
- enable ipv6 rip
- show ipv6 rip

enable ipv6

**Syntax**

ENAble IPV6

**Description**

This command enables the IPv6 module. Any dynamic configuration is preserved between disabling and re-enabling the IPv6 module. The IPv6 module is disabled by default.

**Examples**

To enable IPv6, use the command:

```
en a ipv6
```

**Related Commands**

- disable ipv6
- show ipv6

enable ipv6 advertise

**Description**

This command enables sending of router advertisement packets. To comply with Section 6.2.1 of RFC 2461, IPv6 Neighbour Discovery, the router does not generate router advertisements by default.

As well as using this command to enable advertisements, for each IPv6 interface you want to have advertised, also specify that the interface’s prefix should be included in advertisements. Use one of the commands:

```
ad ipv6 interface=interface ipaddress=(ipv6add/prefix length|dhcp|dhcptemp|pd) publish=yes
```

```
set ipv6 interface=interface publish=yes
```
If the router does not generate router advertisements, the router discovery and address autoconfiguration processes in the network may not work as expected.

If the INTERFACE parameter is specified, sending router advertisements is enabled on that interface. Otherwise, sending router advertisement packets is enabled for all IPv6 interfaces. The interface must already be assigned and configured. Valid interfaces are:

- eth (e.g. eth0)
- PPP (e.g. ppp0)
- VLAN (e.g. vlan1)
- virtual tunnel (e.g. virt9)

To see a list of current valid interfaces, use the show interface command on page 7-66 of Chapter 7, Interfaces, or the show ipv6 interface command on page 15-79.

**Examples**

To enable router advertisements on all interfaces, use the command:

```
en.ipv6 adv
```

**Related Commands**

disable ipv6 advertise

**enable ipv6 debug**

**Syntax**

`enable ipv6 debug`

**Description**

This command enables debugging of IPv6 packets. Debugging is disabled by default.

**Examples**

To enable debugging, use the command:

```
en.ipv6 deb
```

**Related Commands**

disable ipv6 debug

**show ipv6**

**enable ipv6 mtudiscovery**

**Syntax**

`enable ipv6 mtudiscovery`

**Description**

This command enables Path MTU Discovery. This mechanism allows the router to determine the actual maximum transmission units (MTU) of nodes on
possible paths, and use a path with a higher PMTU than the IPv6 minimum link MTU if such a path exists. Path MTU Discovery is disabled by default.

**Examples**
To enable Path MTU Discovery, use the command:

```
en enable ipv6 mtu
```

**Related Commands**
`disable ipv6 mtudiscovery`

## enable ipv6 rip

**Syntax**

```
enable ipv6 rip
```

**Description**
This command enables routing of IPv6 packets using RIPv6. RIPv6 is disabled by default.

**Examples**
To enable the sending and receiving of RIPv6 packets, use the command:

```
en enable ipv6 rip
```

**Related Commands**
```
add ipv6 rip
delete ipv6 rip
disable ipv6 rip
show ipv6 rip
```

## reset ipv6 ndcache

**Syntax**

```
reset ipv6 ndcache
```

**Description**
This command flushes the Neighbour Discovery cache of all entries. Stale ND entries are also automatically flushed every 5 minutes.

**Examples**
To manually purge the Neighbour Discovery cache, use the command:

```
reset ipv6 nd
```

**Related Commands**
```
add ipv6 nd
show ipv6 ndcache
```
set ipv6 filter

Syntax

```
SET IPV6 FILTER=filter-id ENTRY=entry-number
SOURCE=ipv6addr/prefix-length [ACTION={INCLUDE|EXCLUDe}]
PRIOriority=P0..P7] [DESTination=ipv6addr/prefix-length]
[DPORT={port-name|port-id|Any}]
[ICMPPCode={icmp-code-name|icmp-code-id|Any}]
[ICMPPtype={icmp-type-name|icmp-type-id|Any}]
[LOG={4..1950|Dump|Header|None} [OPTIONS={YES|NO|ON|OFF|True|False}]
[PROTocol={protocol|Any|Egp|Icmp|Ospf|Tcp|Udp}]
[SESSION={Any|Established|Start}] [SIZE={size|Any}]
[SPORT={port-name|port-id|Any}]
```

where:

- `filter-id` is an integer from 0 to 99 or 200 to 299.
- `entry-number` is the position of this entry in the filter.
- `ipv6addr` is a valid IPv6 address, with its prefix length optionally indicated by slash notation (see "IPv6 Interfaces and Addresses" on page 15-13).
- `prefix-length` is an integer from 1 to 128.
- `port-name` is the predefined name of a TCP or UDP port.
- `port-id` is an IP port number or a range of port numbers in the format [low]:[high].
- `icmp-code-name` is the predefined name for an ICMP reason code (see Table 15-4 on page 15-37).
- `icmp-code-id` is the number of an ICMP reason code (see Table 15-4 on page 15-37).
- `icmp-type-name` is the predefined name of an ICMP message type (see Table 15-5 on page 15-37).
- `icmp-type-id` is the number of an ICMP message type (see Table 15-5 on page 15-37).
- `protocol` is an IPv6 protocol number.
- `size` is an integer from 0 to 65535.

Description

This command changes an entry in an IP traffic filter or priority filter.

The FILTER parameter specifies the number of the filter in which the entry is to be changed. Filters with numbers from 0 to 99 are treated as traffic filters, and use the ACTION parameter to specify the action to take with a packet that matches the entry. Filters with numbers from 200 to 299 are treated as priority filters, and use the PRIORITY parameter to specify the priority to assign to a packet that matches the entry. An interface may have a maximum of one traffic filter and one priority filter. The same traffic or priority filter can be assigned to more than one interface. Traffic filters are applied to packets received via the interface, whereas priority filters are applied to packets as they are transmitted.

The ENTRY parameter specifies the entry number to be changed.

The ACTION parameter specifies, for traffic filters, the action to take when the entry is matched. If INCLUDE is specified, the IP packet is processed and forwarded. If EXCLUDE is specified, the IP packet is discarded. The ACTION and PRIORITY parameters are mutually exclusive—only one may be specified.
The default is INCLUDE.

The DESTINATION parameter specifies the destination IP address, in IPv6 notation, for the entry. A prefix length can be specified using slash notation (e.g. 3FFE::0/16). The prefix is used to determine the portion of the destination IPv6 address in the IPv6 packet that is significant for comparison with this entry. The default prefix length is 128.

The DPORT parameter specifies the destination port to check against for this entry as the recognised name of a well-known UDP or TCP port, (see Table 15-3 on page 15-36), a decimal value from 0 to 65535, or a range of numbers in the form [low]:[high]. If low is omitted, 0 is assumed. If high is omitted, the maximum port number is assumed. If a port other than ANY is specified, the PROTOCOL parameter must also be specified, and must be one of TCP or UDP. The default is ANY.

The ICMPCODE parameter specifies the ICMP message reason code to match against the ICMP code field in an ICMP packet as a decimal value from 0 to 255, or the recognised name of an ICMP reason code (see Table 15-4 on page 15-37). This parameter is used as a basis for packet matching when the PROTOCOL parameter is set to ICMP. The default is ANY.

The ICMPETYPE parameter specifies the ICMP message type to match against the ICMP type field in an ICMP packet header as a decimal value from 0 to 255 or the recognised name of an ICMP type (see Table 15-5 on page 15-37). This parameter is used as a basis for packet matching when the PROTOCOL parameter is set to ICMP. The default is ANY.

The LOG parameter specifies whether matches to a filter entry result in a log message being sent to the router’s logging facility, and the content of the log messages. This parameter enables logging of the IP packet filtering process down to the level of an individual filter entry.

- If a number from 4 to 1950 is specified, the first 4 to 1950 octets of the data portion of TCP, UDP, and ICMP packets or the first 4 to 1950 octets after the IP header of other protocol packets are logged with a message type/subtype of IPFIL/DUMP. The filter number, entry number and IP header information (source and destination IP addresses, protocol, source and destination ports, and size) are also logged with a message type/subtype of IPFIL/PASS (for entries with an INCLUDE action) or IPFIL/FAIL (for entries with an EXCLUDE action).
- If DUMP is specified, the filter number, entry number and IP header information (source and destination IP addresses, protocol, source and destination ports, and size) are logged with a message type/subtype of IPFIL/PASS (for entries with an INCLUDE action) or IPFIL/FAIL (for entries with an EXCLUDE action). In addition, the first 40 octets of the data portion of TCP, UDP and ICMP packets, or the first 40 octets after the IP header of other protocol packets are logged with a message type/subtype of IPFIL/DUMP.
- If HEADER is specified, the filter number, entry number and IP header information (source and destination IP addresses, protocol, source and destination ports, and size) are logged with a message type/subtype of IPFIL/PASS (for entries with an INCLUDE action) or IPFIL/FAIL (for entries with an EXCLUDE action).
- If NONE is specified, matches to the filter entry are not logged.
- The default is NONE.
The OPTIONS parameter specifies the presence or absence of any time-length-variable (TLV) encoded “Options” to check against. TLV-encoded options can be found in the Hop-by-Hop and Destination Options extension headers. If YES is specified, the entry matches IP packets with any extension header TLV options set. If NO is specified, the entry matches IP packets without any extension header TLV options set. The default is NO.

The PRIORITY parameter specifies, for priority filters, the priority to apply to forwarding packets when the entry is matched. A low value (P0) assigns a high priority to the packet. A high value (P7) assigns a low priority to the packet. The priority number is employed during forwarding (transmission). The default is P7. The ACTION and PRIORITY parameters are mutually exclusive—only one may be specified.

The PROTOCOL parameter specifies a protocol to check against as a decimal value from 0 to 65534, or the recognised name of an IP protocol type. If either the SPORT or DPORT parameters are used, PROTOCOL must be defined as TCP, UDP or ANY. Specifying TCP or UDP filters packets from companion protocols, such as ICMP and OSPF, that do not use TCP or UDP as a transport mechanism. The default is ANY.

The SESSION parameter specifies the type of TCP packet to match and is used as a basis for packet filtering when the PROTOCOL parameter specifies TCP. If START is specified, the entry matches TCP packets with the SYN bit set and the ACK bit clear. If ESTABLISHED is specified, the entry matches TCP packets with either the SYN bit clear or the ACK bit set. If ANY is specified, the entry matches any TCP packet. The default is ANY.

The SIZE parameter specifies the maximum unassembled size to match against for each IP packet or fragment. If the fragment’s offset plus size is greater than the value specified, the fragment is discarded. The default is ANY, which indicates that size is not required as a matching category.

The SOURCE parameter specifies the source IP address, in IPv6 notation, for the entry. A prefix length can be specified using slash notation (e.g. 3ffe::0/16). The prefix is used to determine the portion of the source IPv6 address in the IPv6 packet that is significant for comparison with this entry. The default prefix length is 128.

The SPORT parameter specifies the source port to check against as the recognised name of a well-known UDP or TCP port, (see Table 15-3 on page 15-36), a decimal value from 0 to 65535, or a range of numbers in the form [low]:[high]. If low is omitted, 0 is assumed. If high is omitted, the maximum port number is assumed. If a port other than ANY is specified, the PROTOCOL parameter must also be specified, and must be one of TCP or UDP. The default is ANY.

Examples
To set a filter to monitor and log all Telnet sessions from IP address 3ffe::3, to IP address 3ffe::4, use the command:

```
set ipv6 fil=2 so=3ffe::3/128 des=3ffe::4/64 si=any prot=tcp ac=incl sp=any dp=23 log=h
```

Related Commands
add ipv6 filter
delete ipv6 filter
#### set ipv6 interface

**Syntax**

```plaintext
SET IPV6 INTERFACE=interface [FILTER=0..99]
[IPaddress=ipv6add/prefix-length]
[PREFERRED=1..4294967295|INFinite]
[PRIORITYFILTER=200..299] [PUBLISH=(YES|NO|ON|OFF|True|False)] [VALID=1..4294967295|INFinite]
```

where:
- **interface** is a valid interface.
- **ipv6add** is a valid IPv6 address, with its prefix length indicated by slash notation (see "IPv6 Interfaces and Addresses" on page 15-13).
- **prefix-length** is an integer from 1 to 128.

**Description**

This command modifies various values associated with an interface that was previously created by a call to either the `create ipv6 interface` command on page 15-49, or the `add ipv6 interface` command on page 15-41.

The INTERFACE parameter specifies the physical interface to set the properties of. The interface must already be assigned and configured. Valid interfaces are:
- eth (e.g. eth0)
- PPP (e.g. ppp0)
- VLAN (e.g. vlan1)
- virtual tunnel (e.g. virt9)

To see a list of current valid interfaces, use the `show interface` command on page 7-66 of Chapter 7, Interfaces, or the `show ipv6 interface` command on page 15-79.

The IPADDRESS parameter specifies the IPv6 address to associate with this interface, in IPv6 notation. A prefix length must be specified, using slash notation (e.g. 3FFE::0/16). IPADDRESS is required if the VALID or PREFERRED parameters are specified.

The FILTER parameter is the number of the filter that is to be applied to this interface.

The PREFERRED parameter specifies the time in seconds for which this IPv6 address is the preferred address for this interface. The default is 604800 seconds (7 days). The value of this parameter cannot be greater than that of the VALID parameter. The PREFERRED parameter can be specified if an IPv6 address is specified using the IPADDRESS parameter.

The PRIORITYFILTER parameter is the number of the priority filter attached to this interface.

The PUBLISH parameter determines whether to include the prefix in router advertisement packets. The default is NO. Note that, to comply with Section 6.2.1 of RFC 2461, by default the router does not generate router advertisements. To enable the router to generate router advertisements that include this router’s address, you must specify PUBLISH=YES, and also enable advertisements using the `enable ipv6 advertise` command on page 15-58. If the
router does not generate router advertisements, the router discovery and address autoconfiguration processes in the network may not work as expected.

The VALID parameter is the time in seconds that the IPv6 address exists on the interface. After the time specified expires, the IPv6 address is deleted. The default is 2592000 seconds (30 days). The PREFERRED parameter must also be specified, and the valid time must be the same as or greater than the preferred time. The VALID parameter can be specified if an IPv6 address is specified using the IPADDRESS parameter.

**Examples**
To set the interface address 3ffe::1 to be the preferred address for 300 seconds, use the command:

```
set ipv6 int=eth0 ip=3ffe::1 pref=300
```

**Related Commands**
- add ipv6 interface
- create ipv6 interface
- delete ipv6 interface
- destroy ipv6 interface
- show ipv6
- show ipv6 interface

---

**set ipv6 mtu**

**Syntax**

```
SET IPV6 MTU=mtu INTERFACE=interface
```

where:

- **mtu** is an integer between 1280 and the maximum True MTU of the specified interface.
- **interface** is a valid interface.

**Description**
This command sets the link maximum transmission unit (MTU) on the specified interface. Valid interfaces are:

- eth (e.g. eth0)
- PPP (e.g. ppp0)
- VLAN (e.g. vlan1)
- virtual tunnel (e.g. virt9)

To see a list of current valid interfaces, use the `show interface` command on page 7-66 of Chapter 7, Interfaces, or the `show ipv6 interface` command on page 15-79.

**Examples**
To give the eth0 interface an MTU of 1450, use the command:

```
set ipv6 mtu=1450 int=eth0
```

**Related Commands**
- disable ipv6 mtudiscovery
- enable ipv6 mtudiscovery
- set ipv6 nd
set ipv6 nd

Syntax

```
set ipv6 nd>
```

**SET IPV6 ND**

**INTERface=interface**

[DUPtrans=1..16]

[HOP=1..255]

[LIFE=0|4..9000]

[MAXAInt=4..1800]

[MCONF={YES|NO}]

[MINAInt=3..1350]

[MTU=1280..65535]

[OCONF={YES|NO}]

[REAchie=0..3600000]

[RETRans=0..4294967295]

where interface is a valid interface

Description

This command allows the configuration of various Router Advertisement parameters on the specified IPv6 interface. The interface must already be assigned and configured. Valid interfaces are:

- eth (e.g. eth0)
- PPP (e.g. ppp0)
- VLAN (e.g. vlan1)
- virtual tunnel (e.g. virt9)

To see a list of current valid interfaces, use the `show interface` command on page 7-66 of Chapter 7, Interfaces, or the `show ipv6 interface` command on page 15-79.

The DUPTRANS parameter is the DupAddrDetectTransmits value. It is the number of Neighbour Solicitation messages that the router sends while performing Duplicate Address Detection on a tentative address. The default is 1.

The HOP parameter specifies the AdvCurHopLimit. This value limits the number of hops that packets can take. A value of 0 (zero) indicates that this value is not specified by this router. The default is 64.

The LIFE parameter is the AdvDefaultLifetime in seconds. This is the maximum time that the router is the default. A host sends data to the router that is the current default router in its list of routers. A value of 0 (zero) indicates that the router is not to be used as a default router. If the value is not 0, it must be equal to or greater than MAXAINT and should be 3 * MAXAINT. The default is 1800.

The MAXAINT parameter specifies the MaxRtrAdvInterval, in seconds. This is the maximum time interval between unsolicited multicast Router Advertisements sent by the router from this interface. The default is 600 seconds.

The MCONF parameter is the AdvManaged flag. If TRUE is specified, hosts receiving the advertisements use the administered stateful autoconfiguration protocol for address autoconfiguration, and request address information and other (non-address) information from the router. The default is FALSE.

A change in the AdvManaged flag from TRUE to FALSE in the router’s advertisements is ignored by hosts that are already using stateful address autoconfiguration.

The MINAINT parameter specifies the MinRtrAdvInterval, in seconds. This is the maximum time interval between unsolicited multicast Router
Advertisements sent by the router from this interface. It should be 0.33 * MAXAINT and must be no greater than 0.75 * MAXAINT. The default is 200 seconds.

The MTU parameter is the AdvLinkMTU. This is the maximum transmission unit (MTU) for this link and is used by nodes that are running Path MTU Discovery. The default is 0, which indicates that this value is not specified by this router.

The OCONF parameter is the AdvOtherConfig flag. If TRUE is specified, hosts receiving the advertisements use the administered stateful protocol for autoconfiguration of non-address information. The default is FALSE. If MCONF is TRUE, OCONF implicitly also becomes TRUE.

A change in the AdvOtherConfig flag from TRUE to FALSE in the router’s advertisements is ignored by hosts that are already using stateful address autoconfiguration to obtain non-address information.

The REACH parameter is the AdvReachableTime, in milliseconds. When a node receives a reachability confirmation message as part of the Neighbour Unreachability Detection algorithm, it assumes that the router is reachable for this length of time. The default is 0, which indicates that this timer is not specified by this router.

The RETRANS parameter is the AdvRetransTimer, in milliseconds. This is the interval between repeats of each Router Advertisement message sent by the router. The value you enter is rounded up to the nearest 100 milliseconds (for example, 301 becomes 400). The default is 0, which indicates that this timer is not specified by this router.

**Examples**

To set the router to advertise a maximum transmission unit of 1450 over the eth0 interface, use the command:

```
set ipv6 nd int=eth0 mtu=1450
```

**Related Commands**

- `create ipv6 interface`
- `show ipv6 ndcache`
- `show ipv6 ndconfig`
set ipv6 prefix

**Syntax**

```
SET IPV6 PREFIX=ipv6add/prefix-length INTERFACE=interface
[AUTOnomous={YES|NO}] [ONlink={YES|NO}]
[PREFerred=1..4294967295|INFinite]
[VALid=1..4294967295|INFinite]
```

where:

- `ipv6add` is a valid IPv6 address, with its prefix length indicated by slash notation (see “IPv6 Interfaces and Addresses” on page 15-13).
- `prefix-length` is the number between 1 and 128.
- `interface` is a valid interface.

**Description**

This command modifies the settings for the specified prefix that the router advertises on the specified interface. The prefix is not added to the interface’s IPv6 address. The prefix can assist nodes on that subnet with configuration of their local-link IPv6 addresses. More than one prefix can be included in the router advertisements for a given interface. Valid interfaces are:

- `eth` (e.g. eth0)
- `PPP` (e.g. ppp0)
- `VLAN` (e.g. vlan1)
- `virtual tunnel` (e.g. virt9)

To see a list of current valid interfaces, use the `show interface` command on page 7-66 of Chapter 7, Interfaces, or the `show ipv6 interface` command on page 15-79.

If the PUBLISH parameter is set to YES for an interface by using the `add ipv6 interface` command on page 15-41 or the `set ipv6 interface` command on page 15-64, the prefix specified by the command is advertised as well as the prefix specified with this command.

The PREFIX parameter specifies the IPv6 prefix (e.g. 3ffe::/64) to advertise for this interface.

The AUTONOMOUS parameter specifies whether the prefix is for the same autonomous system (for example, the Internet or an intranet). If it is, hosts use this prefix to configure their addresses over this autonomous system. The default is YES.

The ONLINK parameter specifies whether hosts use this prefix to configure their addresses on the local link. The default is YES.

The PREFERRED parameter specifies the time in seconds for which this prefix is preferred over other prefixes. The prefix no longer appears in router advertisements after this period. The default is 604800 seconds (7 days).

The VALID parameter specifies the time in seconds for which the prefix is valid. The prefix is deleted after this period. The default is 2592000 seconds (30 days).
Examples
To make the prefix 3ffe::/64 the preferred prefix for the next 8 hours (28800 seconds), in router advertisements sent over the eth0 interface, use the command:

```
set ipv6 prefix=3ffe::/64 int=eth0 pref=28800
```

Related Commands
- add ipv6 prefix
- delete ipv6 prefix
- show ipv6 ndcache

**set ipv6 route preference**

**Syntax**
```
SET IPV6 ROUTe PREFERENCE=(DEFault|1..65535) PROTocol=RIP
```

**Description**
This command sets the IP route table preference for routes learned via the specified protocol. When more than one route in the route table matches the destination address in an IPv6 packet, the route with the lowest preference value is used to route the packet. If two or more candidate routes have the same preference, the route with the longest prefix is used. All existing dynamically learned routes in the routing table and new routes added later are updated with the specified preference value.

The PREFERENCE parameter sets the preference for routes learned via the specified protocol. If DEFAULT is specified, the preference reverts to the default for this protocol type.

The PROTOCOL parameter specifies which protocol’s routing table is updated with the new preference value.

**Examples**
To set RIP routes to a lower preference than all OSPF routes, use the command:

```
set ipv6 rou pref=160 prot=rip
```

**Related Commands**
- show ipv6 route preference
show ipv6

Syntax  
Sho IPv6

Description  
This command displays general configuration information for the IPv6 module (Figure 15-15 on page 15-70, Table 15-6 on page 15-70).

Figure 15-15: Example output from the show ipv6 command

```
IPV6 Module Configuration
---------------------------------------------
Module Status .................. Enabled
IPV6 Packet Forwarding ............ Enabled
IPV6 RIP ....................... Disabled
IPV6 Echo Reply .................. Enabled
Name Server .................... 202.49.72.50
Secondary Name Server .......... Not Set
Source-Routed Packets ............ Discarded

Routing Protocols
RIP Neighbours ................. 0

Active Routes:
Static ........................ 5
Interface ........................ 1
Neighbour Discovery ............. 0
RIP ............................ 0
Other .......................... 0
-----------------------------------
Total Number of routes.......... 6
Discarded routes ........... 0
```

Table 15-6: Parameters in the output of the show ipv6 command

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>Module Status</td>
<td>Whether the IPv6 module has been enabled or disabled.</td>
</tr>
<tr>
<td>IPV6 Packet Forwarding</td>
<td>Whether the router is enabled or disabled to forward packets.</td>
</tr>
<tr>
<td>IPV6 RIP</td>
<td>Whether the router has been configured to send and receive RIPv6 packets.</td>
</tr>
<tr>
<td>IPV6 Echo Reply</td>
<td>Whether the router has been configured to reply to echo request (ping) packets.</td>
</tr>
<tr>
<td>Name Server</td>
<td>The IP address of the DNS server for name resolution.</td>
</tr>
<tr>
<td>Secondary Name Server</td>
<td>The IP address of a secondary DNS server, to use when the primary Name Server fails.</td>
</tr>
<tr>
<td>Source-Routed Packets</td>
<td>What the router has been configured to do with source routed packets.</td>
</tr>
<tr>
<td>Routing Protocols</td>
<td>Information about the routing protocols in use.</td>
</tr>
<tr>
<td>RIP Neighbours</td>
<td>The number of nodes on the network known by the router to be running RIPv6.</td>
</tr>
<tr>
<td>Active Routes</td>
<td>Information about the routes in the route table.</td>
</tr>
</tbody>
</table>
### Examples
To show the general settings for the IPv6 module, use the command:
```
sh ipv6
```

### Related Commands
- `show ipv6 counter`
- `show ipv6 filter`
- `show ipv6 host`
- `show ipv6 interface`
- `show ipv6 multicast`
- `show ipv6 ndcache`
- `show ipv6 rip`
- `show ipv6 route`
- `show ipv6 tunnel`

### Table 15-6: Parameters in the output of the show ipv6 command (continued)

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>Static</td>
<td>The number of static routes that have been added to the router through manual configuration.</td>
</tr>
<tr>
<td>Interface</td>
<td>The number of routes that have been generated through an interface being added.</td>
</tr>
<tr>
<td>Neighbour Discovery</td>
<td>The number of routes gathered from neighbour discovery.</td>
</tr>
<tr>
<td>RIP</td>
<td>The number of routes gathered from RIPv6 packets.</td>
</tr>
<tr>
<td>Other</td>
<td>The number of routes gathered through other protocols, for example, OSPF.</td>
</tr>
<tr>
<td>Total Number of Routes</td>
<td>The total number of routes in the route table.</td>
</tr>
<tr>
<td>Discarded Routes</td>
<td>The number of routes that have been discarded because a better route was found through neighbour discovery. Static routes are not discarded.</td>
</tr>
</tbody>
</table>
show ipv6 counter

Syntax

```
SHOW IPV6 COUNTER
```

Description

This command displays the IPv6 MIB counters (Figure 15-16 on page 15-72, Table 15-7 on page 15-72).

Figure 15-16: Example output from the `show ipv6 counter` command

```
IPV6 MIB Counters
------------------------------------------------------------
Interface Counters

Interface: eth0
InReceives ............ 0      OutForwDatagrams ....... 5
InNoRoutes ............ 0      OutRequests ............ 5
InDiscards ............ 0      OutDiscards ............ 0
InAddrErrors .......... 0      OutFragOKs ............. 0
InUnknownProtos ....... 0      OutFragFails ........... 0
InTruncatedPkts ....... 0      OutFragCreates ......... 0
InMcastPkts ........... 0      OutMcastPktcs ........... 0
ReasmReqds ............ 0      ReasmOKs ............... 0
ReasmFails ............ 0
InDelivers ............ 0
InHdrErrors ........... 0
InTooBigErrors ........ 0
ICMP counters

inMsgs ................ 0       OutMsgs ................ 5
inErrors .............. 0       OutErrors .............. 0
inDestUnreachs ........ 0       OutDestUnreachs ........ 0
inAdminProhibs ........ 0       OutAdminProhibs ........ 0
inTimeExcds ........... 0       OutTimeExcds ........... 0
inParmProblems ........ 0       OutParmProblems ........ 0
inPktTooBigs .......... 0       OutPktTooBigs .......... 0
inEchos ............... 0       OutEchos ............... 5
inEchoReplies ......... 0       OutEchoReplies ......... 0
inRouterSolicits ...... 0       OutRouterSolicits ...... 0
inRouterAdvert ......... 0       OutRouterAdvert ......... 0
inNeighborSolicits .... 0       OutNeighborSolicits .... 0
inNeighborAdvert ...... 0       OutNeighborAdvert ...... 0
inRedirects ........... 0       OutRedirects ........... 0
inGroupMembQueries .... 0       OutGroupMembQueries .... 0
inGroupMembResp ....... 0       OutGroupMembResp ....... 0
inGroupMembReduct ..... 0       OutGroupMembReduct ..... 0
```

Table 15-7: Parameters in the output of the `show ipv6 counter` command

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>Interface Counters</td>
<td>Counters for each IPv6 interface on the route.</td>
</tr>
<tr>
<td>Interface</td>
<td>Name of the IPv6 interface.</td>
</tr>
<tr>
<td>InReceives</td>
<td>Number of packets received on the interface.</td>
</tr>
<tr>
<td>InNoRoutes</td>
<td>Number of input packets discarded because no route could be found to transmit them to their destination.</td>
</tr>
<tr>
<td>InDiscards</td>
<td>Number of incoming packets that were discarded.</td>
</tr>
</tbody>
</table>

Software Release 2.7.1
C613-03091-00 REV A
Table 15-7: Parameters in the output of the `show ipv6 counter` command (continued)

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>InAddrErrors</td>
<td>Number of packets received with invalid addresses.</td>
</tr>
<tr>
<td>InUnknownProtos</td>
<td>Number of packets received with unknown next headers.</td>
</tr>
<tr>
<td>InTruncatedPkts</td>
<td>Number of packets received that were truncated.</td>
</tr>
<tr>
<td>InMcastPkts</td>
<td>Number of multicast packets received.</td>
</tr>
<tr>
<td>ReasmReqsds</td>
<td>Number of packets received that required reassembly.</td>
</tr>
<tr>
<td>ReasmFails</td>
<td>Number of packets that could not be reassembled.</td>
</tr>
<tr>
<td>InDelivers</td>
<td>Number of incoming packets that were successfully delivered to a higher layer protocol.</td>
</tr>
<tr>
<td>InHdrErrors</td>
<td>Number of packets received with invalid headers.</td>
</tr>
<tr>
<td>InTooBigErrors</td>
<td>Number of packets received that were discarded because they were too big.</td>
</tr>
<tr>
<td>OutForwDatagrams</td>
<td>Number of packets that have been forwarded.</td>
</tr>
<tr>
<td>OutRequests</td>
<td>Number of echo requests sent.</td>
</tr>
<tr>
<td>OutDiscards</td>
<td>Number of packet discarded messages sent.</td>
</tr>
<tr>
<td>OutFragOKs</td>
<td>Number of fragmentation success messages sent.</td>
</tr>
<tr>
<td>OutFragFails</td>
<td>Number of fragmentation failed messages sent.</td>
</tr>
<tr>
<td>OutFragCreates</td>
<td>Number of outgoing packet fragments generated as a result of fragmentation at this output interface.</td>
</tr>
<tr>
<td>OutMcastPkts</td>
<td>Number of multicast packets sent.</td>
</tr>
<tr>
<td>ReasmOKs</td>
<td>Number of IPv6 packets successfully reassembled.</td>
</tr>
<tr>
<td></td>
<td>Note that this counter is incremented at the interface to which these packets were addressed, which is not necessarily the input interface for some of the fragments.</td>
</tr>
<tr>
<td>ICMP Counters</td>
<td>Counters for ICMPv6 for all IPv6 interfaces on the router.</td>
</tr>
<tr>
<td>inMsgs</td>
<td>Total number of ICMP messages received by the interface, which includes all those counted by InErrors. Note that this interface is the interface to which the ICMP messages were addressed. This may not necessarily be the input interface for the messages.</td>
</tr>
<tr>
<td>InErrors</td>
<td>Number of ICMP messages that the interface received but determined as having ICMP-specific errors (bad ICMP checksums, bad length, etc).</td>
</tr>
<tr>
<td>InDestUnreachs</td>
<td>Number of ICMP Destination Unreachable messages received by the interface.</td>
</tr>
<tr>
<td>InAdminProhibs</td>
<td>Number of ICMP Destination Unreachable/Communication Administratively Prohibited messages received by the interface.</td>
</tr>
<tr>
<td>InTimeExceeds</td>
<td>Number of ICMP Time Exceeded messages received by the interface.</td>
</tr>
<tr>
<td>InParmProblems</td>
<td>Number of ICMP Parameter Problem messages received by the interface.</td>
</tr>
<tr>
<td>InPktTooBigs</td>
<td>Number of ICMP Packet Too Big messages received by the interface.</td>
</tr>
<tr>
<td>InEchoes</td>
<td>Number of ICMP Echo (request) messages received by the interface.</td>
</tr>
<tr>
<td>InEchoReplies</td>
<td>Number of ICMP Echo Reply messages received by the interface.</td>
</tr>
<tr>
<td>Parameter</td>
<td>Description</td>
</tr>
<tr>
<td>----------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>InRouterSolicits</td>
<td>Number of ICMP Router Solicit messages received by the interface.</td>
</tr>
<tr>
<td>InRouterAdvert</td>
<td>Number of ICMP Router Advertisement messages received by the interface.</td>
</tr>
<tr>
<td>InNeighborSolicits</td>
<td>Number of ICMP Neighbor Solicitation messages received by the interface.</td>
</tr>
<tr>
<td>InNeighbor Advert</td>
<td>Number of ICMP Neighbor Advertisement messages received by the interface.</td>
</tr>
<tr>
<td>InRedirects</td>
<td>Number of Redirect messages received by the interface.</td>
</tr>
<tr>
<td>InGroupMembQueries</td>
<td>Number of ICMPv6 Group Membership Query messages received by the interface.</td>
</tr>
<tr>
<td>InGroupMembResp</td>
<td>Number of ICMPv6 Group Membership Response messages received by the interface.</td>
</tr>
<tr>
<td>InGroupMembReduct</td>
<td>Number of ICMPv6 Group Membership Reduction messages received by the interface.</td>
</tr>
<tr>
<td>OutMsgs</td>
<td>Total number of ICMP messages that the interface attempted to send.</td>
</tr>
<tr>
<td></td>
<td>Note that this counter includes all those counted by OutErrors.</td>
</tr>
<tr>
<td>OutErrors</td>
<td>Number of ICMP messages that this interface did not send due to problems discovered within ICMP, such as a lack of buffers. This value does not include errors discovered outside the ICMP layer, such as the inability of IPv6 to route the resulting packet.</td>
</tr>
<tr>
<td>OutDestUnreachs</td>
<td>Number of ICMP Destination Unreachable messages sent by the interface.</td>
</tr>
<tr>
<td>OutAdminProhibs</td>
<td>Number of ICMP destination Unreachable/Communication Administratively Prohibited messages sent by the interface.</td>
</tr>
<tr>
<td>OutTimeExcds</td>
<td>Number of ICMP Time Exceeded messages sent by the interface.</td>
</tr>
<tr>
<td>OutParmProblems</td>
<td>Number of ICMP Parameter Problem messages sent by the interface.</td>
</tr>
<tr>
<td>OutPktTooBigs</td>
<td>Number of ICMP Packet Too Big messages sent by the interface.</td>
</tr>
<tr>
<td>OutEchos</td>
<td>Number of ICMP Echo (request) messages sent by the interface.</td>
</tr>
<tr>
<td>OutEchoREplies</td>
<td>Number of ICMP Echo Reply messages sent by the interface.</td>
</tr>
<tr>
<td>OutRouterSolicits</td>
<td>Number of ICMP Router Solicit messages sent by the interface.</td>
</tr>
<tr>
<td>OutRouterAdvert</td>
<td>Number of ICMP Router Advertisement messages sent by the interface.</td>
</tr>
<tr>
<td>OutNeighborSolicits</td>
<td>Number of ICMP Neighbor Solicit messages sent by the interface.</td>
</tr>
</tbody>
</table>
Table 15-7: Parameters in the output of the `show ipv6 counter` command (continued)

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>OutNeighborAdvert</td>
<td>Number of ICMP Router Advertisement messages sent by the interface.</td>
</tr>
<tr>
<td>OutRedirects</td>
<td>Number of Redirect messages sent by the interface. For a host, this object is always zero since hosts do not send redirects.</td>
</tr>
<tr>
<td>OutGroupMembQueries</td>
<td>Number of ICMPv6 Group Membership Query messages sent by the interface.</td>
</tr>
<tr>
<td>OutGroupMembResp</td>
<td>Number of ICMPv6 Group Membership Response messages sent by the interface.</td>
</tr>
<tr>
<td>OutGroupMembReduct</td>
<td>Number of ICMPv6 Group Membership Reduction messages sent by the interface.</td>
</tr>
</tbody>
</table>

Examples

To show the IPv6 counters, use the command:

```
sh ipv6 cou
```

Related Commands

- show ipv6
- show ipv6 counter
- show ipv6 filter
- show ipv6 host
- show ipv6 interface
- show ipv6 multicast
- show ipv6 ndcache
- show ipv6 rip
- show ipv6 route
- show ipv6 tunnel
show ipv6 filter

Syntax

```
SHOW IPV6 FILTER[=filter-id]
```

where `filter-id` is a number from 0 to 99 or 200 to 299

Description

This command displays information about filters. If a filter is specified, the entries in the filter are displayed. If a filter is not specified, the entries in all filters are displayed (Figure 15-17 on page 15-76, Table 15-8 on page 15-77).

Figure 15-17: Example output from the `show ipv6 filter` command.

<table>
<thead>
<tr>
<th>IP Filters</th>
</tr>
</thead>
<tbody>
<tr>
<td>No.</td>
</tr>
<tr>
<td>-----</td>
</tr>
<tr>
<td>1</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>2</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
</tbody>
</table>

Passes: 0    Fails: 636

|     |      |                |       |             |                     |       |                  |      |          |       |       |         |         |          |         |
|     |      |                |       |             |                     |       |                  |      |          |       |       |         |         |          |         |
|     |      |                |       |             |                     |       |                  |      |          |       |       |         |         |          |         |
|     |      |                |       |             |                     |       |                  |      |          |       |       |         |         |          |         |

Passes: 0    Fails: 0
Internet Protocol Version 6 (IPv6)  
show ipv6 filter  

Table 15-8: Parameters in the output of the **show ipv6 filter** command.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>Source Address</td>
<td>Source IPv6 address.</td>
</tr>
<tr>
<td>Source Port</td>
<td>Source IPv6 port.</td>
</tr>
<tr>
<td>Plen</td>
<td>Number of leftmost contiguous bits of the address to match against.</td>
</tr>
<tr>
<td>Destination Address</td>
<td>Destination IPv6 address.</td>
</tr>
<tr>
<td>Destination Port</td>
<td>Destination IPv6 port.</td>
</tr>
<tr>
<td>Size</td>
<td>Size of the packet to match against.</td>
</tr>
<tr>
<td>Prot. (C/T)</td>
<td>Protocol to match against. If the protocol is ICMP, the ICMP type and code are shown.</td>
</tr>
<tr>
<td>Options</td>
<td>Whether the Hop by Hop or Destination Option extension headers are present.</td>
</tr>
<tr>
<td>Session</td>
<td>Status of the TCP session establishment process.</td>
</tr>
<tr>
<td>Logging</td>
<td>Log details of the packet.</td>
</tr>
<tr>
<td>Matches</td>
<td>Number of packets that matched the entry.</td>
</tr>
<tr>
<td>Act/Pri</td>
<td>Action to take or priority to give if packets match the entry.</td>
</tr>
</tbody>
</table>

**Examples**

To show all filters, use the command:

```
sh ipv6 fil
```

To show all entries in filter 2, use the command:

```
sh ipv6 fil=2
```

**Related Commands**

- **add ipv6 filter**
- **delete ipv6 filter**
- **set ipv6 filter**
### show ipv6 host

**Syntax**
```
show ipv6 host
```

**Description**
This command uses the HOST parameter to display information on the host names associated with IPv6 addresses on the router (Figure 15-18 on page 15-78, Table 15-9 on page 15-78).

![Figure 15-18: Example output from the `show ipv6 host` command.](image)

#### Table 15-9: Parameters in the output of the `show ipv6 host` command.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>IPv6 Address</td>
<td>The IPv6 address of the host.</td>
</tr>
<tr>
<td>Host Name</td>
<td>The alias that has been assigned to that host.</td>
</tr>
</tbody>
</table>

**Examples**
To show the host names associated with the IPv6 addresses on the router, use the command:
```
sh ipv6 ho
```

**Related Commands**
- `add ipv6 host`
- `delete ipv6 host`
show ipv6 interface

Syntax

SHOW IPV6 INTERFACE[=interface]

where interface is a valid interface

Description

This command displays information about the interfaces configured for IPv6. If an interface is specified then information for the specified interface is displayed, otherwise information for all IPv6 interfaces is displayed (Figure 15-19 on page 15-79, Table 15-10 on page 15-80).

Figure 15-19: Example output from the show ipv6 interface command.

```
IPV6 Interface Configuration
---------------------------------------------------------------------
Interface .............................. ppp0
Ipv6 Interface Index ................. 1
Link-layer address ..................... 00-00-00-00-00-00
EUI-64 Interface Identifier .......... 0200CDFFFE01F9C1
IPSec .................................. No
True MTU/Link MTU ...................... 1280/1280
Multicast status ....................... Enabled
Send Router Advertisements ? ........ No
Ipv6 Interface Addresses :
  Int   Addresses                                 PLen    Decrement
  Type     Scope  State       Enabled   Valid      Preferred      Publish
  0     fe80::0200:cdff:fe01:f9c1                 /64     No
        unicast  link   preferred   Yes       infinite   infinite       No
  1     2004::0001                                /64     No
        unicast  global preferred   Yes       infinite   infinite       Yes

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

IPV6 Interface Configuration
---------------------------------------------------------------------
Interface .............................. eth0
Ipv6 Interface Index ................. 2
Link-layer address ..................... 00-00-cd-01-f9-c1
EUI-64 Interface Identifier .......... 0200CDFFFE01F9C1
IPSec .................................. No
True MTU/Link MTU ...................... 1500/1500
Multicast status ....................... Enabled
Send Router Advertisements ? ........ No
Ipv6 Interface Addresses :
  Int   Addresses                                 PLen    Decrement
  Type     Scope  State       Enabled   Valid      Preferred      Publish
  0     fe80::0200:cdff:fe01:f9c1                 /64     No
        unicast  link   preferred   Yes       infinite   infinite       No
  1     2001::0001                                /64     No
        unicast  global preferred   Yes       infinite   infinite       Yes
```
Examples
To display information about the eth0 interface, use the command:

```
sh ipv6 int=eth0
```

Related Commands
- `add ipv6 interface`
- `delete ipv6 interface`
- `set ipv6 interface`
- `show ipv6`
show ipv6 multicast

Syntax
SHOW IPV6 MULTICAST

Description
This command displays information about the multicast memberships on the router (Figure 15-20 on page 15-81, Table 15-11 on page 15-81).

Figure 15-20: Example output from the `show ipv6 multicast` command.

Table 15-11: Parameters in the output of the `show ipv6 multicast` command.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>Multicast Address</td>
<td>The address of a multicast group.</td>
</tr>
<tr>
<td>Interface</td>
<td>The interface that belongs to the multicast group.</td>
</tr>
</tbody>
</table>

Examples
To show the multicast settings, use the command:

`sh ipv6 mul`

Related Commands
`show ipv6`
show ipv6 ndcache

Syntax
SHOW IPV6 NDCAche

Description
This command displays information gathered through neighbour discovery (Figure 15-21 on page 15-82, Table 15-12 on page 15-82).

Figure 15-21: Example output from show ipv6 ndcache command

<table>
<thead>
<tr>
<th>IPv6 Neighbour Cache:</th>
<th>Link-layer address</th>
</tr>
</thead>
<tbody>
<tr>
<td>IPv6 Address</td>
<td>State</td>
</tr>
<tr>
<td>Interface</td>
<td></td>
</tr>
<tr>
<td>3ffe:0c00:8017:0120:0220:35ff:feb1:7065</td>
<td>stale</td>
</tr>
<tr>
<td>eth0</td>
<td></td>
</tr>
<tr>
<td>fe80::0220:35ff:feb1:7065</td>
<td>stale</td>
</tr>
<tr>
<td>eth0</td>
<td></td>
</tr>
</tbody>
</table>

Table 15-12: Parameters in the output of the show ipv6 ndcache command

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>IPv6 Address</td>
<td>IPv6 address of the neighbour.</td>
</tr>
<tr>
<td>Link-layer address</td>
<td>Link-layer address of the neighbour.</td>
</tr>
<tr>
<td>Interface</td>
<td>The name of the interface through which the neighbour is reachable.</td>
</tr>
<tr>
<td>State</td>
<td>Whether the neighbour discovery is reachable, unreachable, or stale.</td>
</tr>
<tr>
<td>LastReachable</td>
<td>Number of milliseconds within which the neighbour was last reachable.</td>
</tr>
<tr>
<td>IsRouter</td>
<td>Whether the neighbour is a router.</td>
</tr>
</tbody>
</table>

Examples
To show neighbour information, use the command:

```
sh ipv6 ndca
```

Related Commands
disable ipv6
enable ipv6
show ipv6 ndconfig

Syntax

Show IPv6 NDConfig [INterface=interface]

where interface is a valid interface

Description

This command displays various Neighbour Discovery parameters. If an IPv6 interface is specified, only parameters on that interface are displayed. Valid interfaces are:

- eth (e.g. eth0)
- PPP (e.g. ppp0)
- VLAN (e.g. vlan1)
- virtual tunnel (e.g. virt9)

Figure 15-22: Example output from the show ipv6 ndconfig command

Table 15-13: Parameters in the output of the show ipv6 ndconfig command

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>Interface</td>
<td>Name of the IPv6 interface to the neighbour.</td>
</tr>
<tr>
<td>AdvSendAdvertisements</td>
<td>Whether the router sends Router Advertisements.</td>
</tr>
<tr>
<td>MaxRtrAdvInterval</td>
<td>Maximum time interval between unsolicited multicast Router Advertisements sent by the router, in seconds.</td>
</tr>
<tr>
<td>MinRtrAdvInterval</td>
<td>Minimum time in seconds between unsolicited multicast Router Advertisements sent by the router.</td>
</tr>
<tr>
<td>AdvManagedFlag</td>
<td>Whether hosts receiving the advertisements use the administered stateful autoconfiguration protocol for address autoconfiguration.</td>
</tr>
<tr>
<td>AdvOtherConfigFlag</td>
<td>Whether hosts receiving the advertisements use the administered stateful autoconfiguration protocol for autoconfiguration of non-address information.</td>
</tr>
<tr>
<td>AdvLinkMTU</td>
<td>Maximum transmission unit (MTU) for this link.</td>
</tr>
<tr>
<td>AdvReachableTime</td>
<td>Time in seconds after receiving the Router Advertisement that a node assumes the router is reachable.</td>
</tr>
</tbody>
</table>
Table 15-13: Parameters in the output of the `show ipv6 ndconfig` command (continued)

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>AdvRetransTimer</td>
<td>The interval in milliseconds between repeats of each Router Advertisement message sent by the router.</td>
</tr>
<tr>
<td>AdvCurHopLimit</td>
<td>Current Hop Limit, which is used to limit the number of hops that packets can take.</td>
</tr>
<tr>
<td>AdvDefaultLifetime</td>
<td>Maximum time in seconds that the router is the default router.</td>
</tr>
<tr>
<td>AdvPrefixList</td>
<td>Prefixes that are included in the Router Advertisements.</td>
</tr>
<tr>
<td>plen</td>
<td>The prefix length in bits.</td>
</tr>
<tr>
<td>valid</td>
<td>How long in seconds that the prefix is valid. The prefix is deleted after this period.</td>
</tr>
<tr>
<td>pref</td>
<td>How long in seconds that the preferred prefix is valid. The prefix is no longer advertised after this period.</td>
</tr>
<tr>
<td>A</td>
<td>Whether the prefix is for the same autonomous system as the neighbour (for example, the Internet or an intranet).</td>
</tr>
<tr>
<td>O</td>
<td>Whether the prefix is on the same link as the neighbour.</td>
</tr>
</tbody>
</table>

**Example**

To display neighbour discovery parameters for the eth0 interface

```
sh ipv6 ndco int=eth0
```

**Related Commands**

- `add ipv6 prefix`
- `set ipv6 nd`
- `set ipv6 prefix`
show ipv6 rip

Syntax

\texttt{SHow IPV6 RIP \{\texttt{COnter|TIMer}\}}

Description

This command displays information about RIP interfaces. If neither COUNTER or TIMER is specified, summary information is displayed about the interfaces on which RIP is enabled (Figure 15-23 on page 15-85, Table 15-14 on page 15-85). If TIMER is specified, information about the timers for each route is displayed (Figure 15-24 on page 15-85, Table 15-15 on page 15-85). If COUNTER is specified, counters for RIP are displayed (Figure 15-25 on page 15-86, Table 15-16 on page 15-86).

Figure 15-23: Example output from the \texttt{show ipv6 rip} command

<table>
<thead>
<tr>
<th>Interface</th>
<th>Poison</th>
<th>IPv6 Address</th>
</tr>
</thead>
<tbody>
<tr>
<td>virt0</td>
<td>Yes</td>
<td>3ffe::0002</td>
</tr>
</tbody>
</table>

Table 15-14: Parameters in the output of the \texttt{show ipv6 rip} command

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>Interface</td>
<td>Name of the IPv6 interface over which RIP is running.</td>
</tr>
<tr>
<td>Poison</td>
<td>Whether poison reverse is enabled on the interface.</td>
</tr>
<tr>
<td>IPv6 Address</td>
<td>IPv6 address of the interface.</td>
</tr>
</tbody>
</table>

Figure 15-24: Example output from the \texttt{show ipv6 rip timer} command

\begin{verbatim}
RIPng route timers:

<table>
<thead>
<tr>
<th>Destination</th>
<th>int.</th>
<th>met.</th>
<th>val</th>
<th>hold</th>
<th>flush</th>
</tr>
</thead>
<tbody>
<tr>
<td>2ffe::/32</td>
<td>1</td>
<td>2</td>
<td>18</td>
<td>0</td>
<td>28</td>
</tr>
<tr>
<td>2ffa::/32</td>
<td>1</td>
<td>2</td>
<td>18</td>
<td>0</td>
<td>28</td>
</tr>
<tr>
<td>2ffb::/32</td>
<td>1</td>
<td>2</td>
<td>18</td>
<td>0</td>
<td>28</td>
</tr>
</tbody>
</table>
\end{verbatim}

Table 15-15: Parameters in the output of the \texttt{show ipv6 rip timer} command

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>Destination</td>
<td>Destination network for the route.</td>
</tr>
<tr>
<td>int</td>
<td>Name of the interface over which RIP is running.</td>
</tr>
<tr>
<td>met</td>
<td>RIP metric associated with this route.</td>
</tr>
<tr>
<td>val</td>
<td>Valid lifetime of the route, in seconds.</td>
</tr>
<tr>
<td>hold</td>
<td>Interval in seconds after the route becomes invalid, during which the router ignores updates for the route that would normally make the route valid again.</td>
</tr>
<tr>
<td>flush</td>
<td>Interval in seconds from the last update of the route, until the route is flushed from the route table.</td>
</tr>
</tbody>
</table>
Figure 15-25: Example output from the `show ipv6 rip counter` command

```
IPV6 RIPng Counter Summary:
Input:
inResponses .......... 3
inDiscards .......... 0
Output:
outResponses .......... 0
```

Table 15-16: Parameters in the output of the `show ipv6 rip counter` command

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>inResponses</td>
<td>Number of RIP packets received.</td>
</tr>
<tr>
<td>inDiscards</td>
<td>Number of RIP packets that were received and discarded.</td>
</tr>
<tr>
<td>outResponses</td>
<td>Number of RIP packets sent.</td>
</tr>
</tbody>
</table>

Examples

To display a list of RIP interfaces, use the command:

```
sh ipv6 rip
```

To display a list of RIP timers, use the command:

```
sh ipv6 rip tim
```

To display a list of RIP counters, use the command:

```
sh ipv6 rip cou
```

Related Commands

- `add ipv6 rip`
- `delete ipv6 rip`
- `disable ipv6 rip`
- `enable ipv6 rip`
- `show ipv6`
show ipv6 route

**Syntax**
```
SHOW IPV6 ROUTE
```

**Description**
This command displays the contents of the IPv6 route table (Figure 15-26 on page 15-87, Table 15-17 on page 15-87).

Figure 15-26: Example output from the `show ipv6 route` command

<table>
<thead>
<tr>
<th>Destination prefix</th>
<th>Next Hop</th>
<th>Int.</th>
<th>Age</th>
<th>Policy</th>
<th>Protocol</th>
<th>Metric</th>
<th>Pref</th>
<th>Tunnel</th>
<th>Flags</th>
</tr>
</thead>
<tbody>
<tr>
<td>2ffe::/32</td>
<td>3ffe::0002</td>
<td>virt0</td>
<td>no</td>
<td>0</td>
<td>static</td>
<td>2</td>
<td>60</td>
<td>yes</td>
<td>-</td>
</tr>
<tr>
<td>2fff::/32</td>
<td>3ffe::0001</td>
<td>virt0</td>
<td>yes</td>
<td>0</td>
<td>ripng</td>
<td>2</td>
<td>100</td>
<td>yes</td>
<td>-</td>
</tr>
<tr>
<td>2ffa::/32</td>
<td>3ffe::0001</td>
<td>virt0</td>
<td>yes</td>
<td>0</td>
<td>ripng</td>
<td>2</td>
<td>100</td>
<td>yes</td>
<td>-</td>
</tr>
<tr>
<td>2ffb::/32</td>
<td>3ffe::0001</td>
<td>virt0</td>
<td>yes</td>
<td>0</td>
<td>ripng</td>
<td>2</td>
<td>100</td>
<td>yes</td>
<td>-</td>
</tr>
</tbody>
</table>

Codes: P=publish, D=default, A=addrconf, S=stale, L=onlink, N=nonexthop, C=cache, F=flow, U=unknown

Table 15-17: Parameters in the output of the `show ipv6 route` command

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>Destination prefix</td>
<td>The destination network for the route.</td>
</tr>
<tr>
<td>Next Hop</td>
<td>The next node along the path to the destination.</td>
</tr>
<tr>
<td>Int.</td>
<td>The interface via which packets are sent.</td>
</tr>
<tr>
<td>Age</td>
<td>Whether the route ages out.</td>
</tr>
<tr>
<td>Policy</td>
<td>Reserved for future use.</td>
</tr>
<tr>
<td>Protocol</td>
<td>The protocol where the route was gathered: static - the route was configured manually via the command line interface - the route was automatically created through adding an interface ripng - the route was determined via RIPv6</td>
</tr>
<tr>
<td>Metric</td>
<td>The RIP metric that is being used. This usually indicates how far away the destination is, in hops.</td>
</tr>
<tr>
<td>Pref</td>
<td>The preference of the route over others to the same destination.</td>
</tr>
<tr>
<td>Tunnel</td>
<td>Whether the interface is a tunnel; one of “yes” or “no”.</td>
</tr>
<tr>
<td>Flags</td>
<td>The various flags that have been set or implied on the route.</td>
</tr>
</tbody>
</table>

**Examples**
To show the router’s IPv6 routes, use the command:
```
sh ipv6 route
```

**Related Commands**
`add ipv6 route`
`delete ipv6 route`
show ipv6 route multicast

**Syntax**

```
SHow IPV6 ROUTe MULTicat[=ipv6address]
[SOurce=ipv6address]
```

where `ipv6address` is an IPv6 address, with or without the wildcard character (`*`). Examples of valid addresses are given below. The wildcard cannot be used in addresses in other IPv6 commands.

**Description**

This command displays information about the IPv6 multicast forwarding table. If no optional parameters are specified, the contents of the forwarding table are displayed in full (Figure 15-27 on page 15-88, Table 15-18 on page 15-89).

If a multicast group address is specified in the MULTICAST parameter, all packet sources that send to the multicast group address are displayed, as well as upstream and downstream interfaces.

If the SOURCE parameter is specified, then multicast groups sourced from the specified IP address are displayed.

In both the MULTICAST and SOURCE parameters, the wildcard character “*” may replace any complete octet in the address, but not part of an octet. The following examples are valid uses of the wildcard:

```
```

If both octets in a pair are to be wildcards, a single wildcard character can be used, so the following two addresses are equivalent:

```
```

An address can contain as many wildcards as required, but must conform to the IPv6 address syntax convention of 8 pairs of octets separated by colons:

```
*:*:1111::
```

Figure 15-27: Example output from the `show ipv6 route multicast` command

<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Source/Prefix</td>
<td>1111:1111/128</td>
</tr>
<tr>
<td>Scope</td>
<td>Site</td>
</tr>
<tr>
<td>Protocol</td>
<td>PIM</td>
</tr>
<tr>
<td>InPort</td>
<td>ppp0</td>
</tr>
<tr>
<td>OutPorts</td>
<td>eth0, eth1</td>
</tr>
<tr>
<td>Source/Prefix</td>
<td>Global</td>
</tr>
<tr>
<td>Protocol</td>
<td>PIM</td>
</tr>
<tr>
<td>InPort</td>
<td>ppp0</td>
</tr>
<tr>
<td>OutPorts</td>
<td>eth1</td>
</tr>
<tr>
<td>Source/Prefix</td>
<td>1111:1111/128</td>
</tr>
<tr>
<td>Scope</td>
<td>Site</td>
</tr>
<tr>
<td>Protocol</td>
<td>PIM</td>
</tr>
<tr>
<td>InPort</td>
<td>eth0</td>
</tr>
<tr>
<td>OutPorts</td>
<td>ppp2</td>
</tr>
</tbody>
</table>
Table 15-18: Parameters in the output of the `show ipv6 route multicast` command

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>Source/Prefix</td>
<td>IPv6 address of the host that is sourcing multicast datagrams addressed to the specified groups.</td>
</tr>
<tr>
<td>Multicast Address</td>
<td>IPv6 address to which multicast datagrams are addressed.</td>
</tr>
<tr>
<td>Scope</td>
<td>Whether the multicast address is set for site or global.</td>
</tr>
<tr>
<td>Protocol</td>
<td>Multicast routing protocol that contributes this forwarding entry.</td>
</tr>
<tr>
<td>InPort</td>
<td>Incoming (upstream) port for the (source, group) pair.</td>
</tr>
<tr>
<td>OutPorts</td>
<td>Outgoing (downstream) ports over which multicast datagrams for the (source, group) pair are forwarded.</td>
</tr>
</tbody>
</table>

**Examples**

To display the complete IPv6 multicast forwarding table, use the command:

```
sh ipv6 rou mul
```

To display the IPv6 multicast forwarding table for packet sources that send to all groups that begin with ff05 and have 2a as the 10th octet, use the command:

```
sh ipv6 rou mul=ff05::*:*:*:*:*:*:*2a::*:*:*:*
```

**Related Commands**

`show ipv6`
**show ipv6 route preference**

**Syntax**
```
show ipv6 route preference
```

**Description**
This command displays information about the current IP route table preferences for the routing protocol. See Figure 15-28 on page 15-90 and Table 15-19 on page 15-90.

Figure 15-28: Example output from the `show ipv6 route preference` command

```
IPv6 Route Preference
-----------------------------------------------
Protocol                  Preference
-----------------------------------------------
RIP                         100 (default)
-----------------------------------------------
```

Table 15-19: Parameters in the output of the `show ipv6 route preference` command

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>Protocol</td>
<td>Available routing protocols</td>
</tr>
<tr>
<td>Preference</td>
<td>Preference value for the routing protocol - a larger preference value indicates a less desirable routing protocol</td>
</tr>
</tbody>
</table>

**Related Commands**
`set ipv6 route preference`
show ipv6 tunnel

**Syntax**
```sh
show ipv6 tunnel
```

**Description**
This command displays information about the tunnels that are configured on the router (Figure 15-29 on page 15-91, Table 15-20 on page 15-91).

Figure 15-29: Example output from the `show ipv6 tunnel` command

```
<table>
<thead>
<tr>
<th>Interface</th>
<th>IPv6 Tunnel Address</th>
<th>Tunnel start</th>
<th>Tunnel end</th>
</tr>
</thead>
<tbody>
<tr>
<td>virt0</td>
<td>fe80::c0a8:0102</td>
<td>192.168.1.2</td>
<td>192.168.1.1</td>
</tr>
</tbody>
</table>
```

Table 15-20: Parameters in the `show ipv6 tunnel` command

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>Interface</td>
<td>Interface of the tunnel. Tunnels always have an interface name of <code>virt</code>. Unless an interface number was specified when the tunnel was created, they are numbered from 0 in order of creation.</td>
</tr>
<tr>
<td>IPv6 Tunnel Address</td>
<td>IPv6 address that the router assigned to the tunnel interface when the tunnel was created.</td>
</tr>
<tr>
<td>Tunnel Start</td>
<td>IPv4 address of the start of the tunnel.</td>
</tr>
<tr>
<td>Tunnel End</td>
<td>IPv4 address of the end of the tunnel.</td>
</tr>
</tbody>
</table>

**Examples**
To show the tunnels configured on the router, use the command:
```
sh ipv6 tun
```

**Related Commands**
- `add ipv6 tunnel`
- `delete ipv6 tunnel`