# Foundry IPv6 Configuration Guide


4980 Great America Parkway
Santa Clara, CA 95054
Tel 408.207.1700

November 2007

# Contents

# Chapter 1
# Getting Started

This guide describes the IPv6 IronWare software and features from Foundry Networks. It provides conceptual information about IPv6 addressing and explains how to configure basic IPv6 connectivity and the IPv6 routing protocols. The software procedures explain how to perform tasks using the CLI. Foundry devices to which this manual applies are listed in Table 1.1 on page 1-3.

## Audience

This manual is designed for system administrators with a working knowledge of Layer 2 and Layer 3 switching and routing.

If you are using a Foundry Layer 3 Switch that supports IPv6, you should be familiar with the following protocols if applicable to your network – IPv6, RIPng, OSPFv3, IPv6 IS-IS, BGP4+, and IPv6 MBGP.

## Nomenclature

This guide uses the following typographical conventions to show information:

| | |
|---|---|
| *Italic* | highlights the title of another publication and occasionally emphasizes a word or phrase. |
| **Bold** | highlights a CLI command. |
| ***Bold Italic*** | highlights a term that is being defined. |
| Underline | highlights a link on the Web management interface. |
| Capitals | highlights field names and buttons that appear in the Web management interface. |

**NOTE:** A note emphasizes an important fact or calls your attention to a dependency.

**WARNING:** A warning calls your attention to a possible hazard that can cause injury or death.

**CAUTION:** A caution calls your attention to a possible hazard that can damage equipment.

# Related Publications

The following Foundry Networks documents supplement the information in this guide.

- *Foundry Switch and Router Installation and Basic Configuration Guide* – provides configuration guidelines for Layer 2 and Layer 3 devices and installation procedures for the Foundry devices with IronCore and JetCore modules, as well as Terathon and FastIron Edge Switch devices.
- *Foundry Security Guide* – provides procedures for securing management access to Foundry devices and for protecting against Denial of Service (DoS) attacks.
- *Foundry Enterprise Configuration and Management Guide* – provides configuration information for enterprise routing protocols including IP, RIP, IP multicast, OSPF, BGP4, VRRP and VRRPE. This guide applies to Foundry devices with IronCore and JetCore modules, as well as Terathon and FastIron Edge Switch devices.
- *Foundry NetIron Service Provider Configuration and Management Guide* – provides configuration information for IS-IS and MPLS for Foundry devices with IronCore and JetCore modules that support IS-IS and MPLS.
- *Foundry NetIron IMR 640 Service Provider Configuration and Management Guide* – provides configuration information for IS-IS and MPLS for for the NetIron IMR 640.
- *Foundry Switch and Router Command Line Interface Reference* – provides a list and syntax information for Foundry devices with IronCore and JetCore modules, as well as Terathon and FastIron Edge Switch devices.
- *Foundry Diagnostic Guide* – provides descriptions of diagnostic commands that can help you diagnose and solve issues on IronCore, JetCore, and Terathon Layer 2 Switches and Layer 3 Switches.
- *Foundry BigIron Mg8 Switch Installation and Basic Configuration Guide* – provides installation procedures for the BigIron MG8. This guide also presents the management modules available in the device.
- *Foundry NetIron 40G Switch Installation and Basic Configuration Guide* – provides installation procedures for the BigIron MG8. This guide also presents the management modules available in the device.
- *NetIron IMR 640 Installation and Basic Configuration Guide* – provides procedures for installing modules into and connecting your DC power source(s) to the NetIron IMR 640 chassis, cabling the Ethernet interface ports, and performing a basic configuration of the software.
- *Foundry Management Information Base Reference* – presents the Simple Network Management Protocol (SNMP) Management Information Base (MIB) objects that are supported in the Foundry devices.
- *Foundry IPv6 Configuration Guide* – provide configuration information for IPv6 features on Foundry devices with IronCore and JetCore modules, as well as Terathon and FastIron Edge Switch devices.
- *Foundry IronPoint Wireless LAN Configuration Guide* – presents the features for the IronPoint wireless LAN (WLAN), which is supported on the IronPoint-FastIron Edge Switch.

# Updates to Manuals

Manuals for this product may be updated between releases. For the latest edition of manuals, check the Foundry Knowledge Portal at kp.foundrynet.com.

# How to Get Help or Report Errors

Foundry Networks is committed to ensuring that your investment in our products remains cost-effective. If you need assistance, or find errors in the manuals, contact Foundry Networks using one of the following options:

## Web Access

Go to kp.foundrynet.com and log in to the Knowledge Portal (KP) to obtain more information about a product, or to report documentation errors. **To report errors, click on Cases > Create a New Ticket.**

## E-mail Access

Sent and e-mail to support@foundrynet.com

### Telephone Access

1.877.TURBOCALL (887.2622) United States

1.408-207-1600 Outside the United States

## Warranty Coverage

Contact Foundry Networks using any of the methods listed above for information about the standard and extended warranties.

## Feature Support

Table 1.1 shows which IPV6 features are supported in devices running IPV6 software.

**NOTE:** On BigIron MG8 and NetIron 40G devices, IPv6 is supported only on High Value Interface modules.

Also, FES devices support IPv6 management features only. These devices do not support IPv6 routing.

**Table 1.1: IPv6 Feature Support on Foundry Devices**

| Features and Requirements | NetIron 4802 Stackable device | NetIron 400/ 800/1500 Chassis devices | BigIron/NetIron Chassis devices | BigIron MG8 and NetIron 40G | FastIron Edge Switch |
|---|---|---|---|---|---|
| Management Module Required | n/a | IronCore VM1 | JetCore module | 2-port and 4-port 10 Gigabit Ethernet modules<br>40-port 1 Gigabit Ethernet module | n/a |
| Minimum Software Version Required | IPv6 IronWare software 01.0.00 | IPv6 IronWare software 02.0.00 | IronWare software 07.7.02 | Terathon IronWare software 02.0.00 | IronWare software 03.4.01 |
| **Basic IPv6 Connectivity** | | | | | |
| Global or Site-Local IPv6 Address | Yes | Yes | Yes | Yes | No |
| Link-Local IPv6 Address | Yes | Yes | Yes | Yes | Yes |
| ANYCAST ADDRESSES | Yes, requires IPv6 IronWare software 02.0.00 | Yes | Yes | Yes | No |
| AAA DNS Resolver | Yes, requires IPv6 IronWare software 02.0.00 | Yes | Yes | Yes | No |

**Table 1.1: IPv6 Feature Support on Foundry Devices**

| Features and Requirements | NetIron 4802 Stackable device | NetIron 400/ 800/1500 Chassis devices | BigIron/NetIron Chassis devices | BigIron MG8 and NetIron 40G | FastIron Edge Switch |
|---|---|---|---|---|---|
| Trunk Group Load Sharing | Yes, requires IPv6 IronWare software 02.0.00 | Yes | Yes | Yes | No |
| ECMP Load Sharing for IPv6 | Yes, requires IPv6 IronWare software 02.0.00 | Yes | Yes | Yes | No |
| **ICMP** | | | | | |
| ICMP Rate Limiting | Yes | Yes | Yes | Yes | No |
| ICMP Redirect Messages | Yes | Yes | Yes | Yes | No |
| Neighbor Discovery | | | | | |
| Neighbor Solicitation and Advertisement | Yes | Yes | Yes | Yes | No |
| Router Advertisement and Solicitation | Yes | Yes | Yes | Yes | No |
| IPv6 MTU | Yes | Yes | Yes | Yes | No |
| Static Neighbor Entries | Yes | Yes | Yes | Yes | No |
| Hop count limit | Yes | Yes | Yes | Yes | No |
| QoS for IPv6 Traffic | Yes | Yes | Yes | Yes | No |
| Clearing IPv6 information | Yes, clearing session flows requires IPv6 IronWare software 02.0.00 | Yes | Yes | Yes | Yes |
| **Static IPv6 Routes** | Yes | Yes | Yes | Yes | No |
| **RIPng** | Yes | Yes | Yes | Yes | No |
| **OSPF Version 3** | Yes | Yes | Yes | Yes | No |
| Configurable LSA pacing interval | Yes, requires IPv6 IronWare software 02.0.00 | Yes | Yes | Yes | No |

**Table 1.1: IPv6 Feature Support on Foundry Devices**

| Features and Requirements | NetIron 4802 Stackable device | NetIron 400/ 800/1500 Chassis devices | BigIron/NetIron Chassis devices | BigIron MG8 and NetIron 40G | FastIron Edge Switch |
|---|---|---|---|---|---|
| Filtering OSPF routes | Yes, requires IPv6 IronWare software 02.0.00 | Yes | Yes | Yes | No |
| **IPv6 IS-IS** | Yes, full SPF calculation requires IPv6 IronWare software 02.0.00 | Yes | Yes | Yes<br>Release 02.2.01 and later | No |
| **BGP4+** | Yes | Yes | Yes | Yes | No |
| **IPv4-to-IPv6 Transition Mechanisms** | | | | | |
| Dual Stack Backbone | Yes | Yes | Yes | Yes | No |
| **IPv6 Access Control List** | Yes. Extended ACL support requires IPv6 IronWare software 02.0.00 | Yes | Yes | Yes | Management only |
| **IPv6 Prefix List** | Yes | Yes | Yes | Yes | No |
| **IPV6 Multicast** | | | | | No |
| **MLD** | No | No | No | Yes | No |
| Source Specific Multicast Protocols | No | No | No | Yes | No |
| **sFlow** | Yes, requires IPv6 IronWare software 02.0.00 | Yes | Yes | Yes.<br>BigIron MG8 requires release 02.02.<br>NetIron 40Grequires software 02.0.01 | No |

**Table 1.1: IPv6 Feature Support on Foundry Devices**

| Features and Requirements | NetIron 4802 Stackable device | NetIron 400/ 800/1500 Chassis devices | BigIron/NetIron Chassis devices | BigIron MG8 and NetIron 40G | FastIron Edge Switch |
|---|---|---|---|---|---|
| **Trunk Server for IPv6** | | | Yes. requires Enterprise release 08.0.00 | Yes.<br>BigIron MG8 requires release 02.02.<br>NetIron 40Grequires software 02.0.01 | No |
| **Device Management Commands** | | | | | |
| IPv6 copy | Yes | Yes | Yes | Yes | Yes |
| IPv6 ncopy | Yes | Yes | Yes | Yes | Yes |
| IPv6 debug | Yes | Yes | Yes | Yes | Yes |
| IPv6 ping | Yes | Yes | Yes | Yes | Yes |
| IPv6 traceroute | Yes | Yes | Yes | Yes | Yes |
| DNS server name resolution | Yes | Yes<br>Limited support for BigIron release 08.0.00 | Yes<br>Limited support for BigIron release 08.0.00 | Yes | Yes |
| HTTP/HTTPS Connections | Yes | Yes<br>Not supported for BigIron release 08.0.00 | Yes<br>Not supported for BigIron release 08.0.00 | Yes | Yes |
| Logging (Syslog) | Yes | Yes<br>Not supported for BigIron release 08.0.00 | Yes<br>Not supported for BigIron release 08.0.00 | Yes | Yes |
| Telnet | Yes | Yes<br>Not supported for BigIron release 08.0.00 | Yes<br>Not supported for BigIron release 08.0.00 | Yes | Yes |
| TFTP | Yes | Yes | Yes | Yes | Yes |
| Secure Shell | Yes | Yes | Yes | Yes | Yes |

**Table 1.1: IPv6 Feature Support on Foundry Devices**

| Features and Requirements | NetIron 4802 Stackable device | NetIron 400/ 800/1500 Chassis devices | BigIron/NetIron Chassis devices | BigIron MG8 and NetIron 40G | FastIron Edge Switch |
|---|---|---|---|---|---|
| SNMP | Yes | Yes<br>Not supported for BigIron release 08.0.00 | Yes<br>Not supported for BigIron release 08.0.00 | Yes | Yes |

# Chapter 2
# IPv6 Addressing Overview

This chapter includes overview information about the following topics:

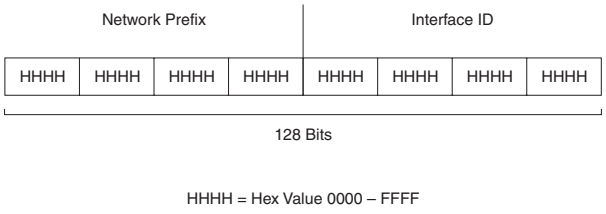
- IPv6 addressing.
- The IPv6 stateless autoconfiguration feature, which enables a host on a local link to automatically configure its interfaces with new and globally unique IPv6 addresses associated with its location.

## IPv6 Addressing

A limitation of IPv4 is its 32-bit addressing format, which is unable to satisfy potential increases in the number of users, geographical needs, and emerging applications. To address this limitation, IPv6 introduces a new 128-bit addressing format.

An IPv6 address is composed of 8 fields of 16-bit hexadecimal values separated by colons (:). Figure 2.1 shows the IPv6 address format.

**Figure 2.1IPv6 address format**




As shown in Figure 2.1, HHHH is a 16-bit hexadecimal value, while H is a 4-bit hexadecimal value. The following is an example of an IPv6 address:

2001:0000:0000:0200:002D:D0FF:FE48:4672

Note that the sample IPv6 address includes hexadecimal fields of zeros. To make the address less cumbersome, you can do the following:

- Omit the leading zeros; for example, 2001:0:0:200:2D:D0FF:FE48:4672.
- Compress the successive groups of zeros at the beginning, middle, or end of an IPv6 address to two colons (::) once per address; for example, 2001::200:2D:D0FF:FE48:4672.

When specifying an IPv6 address in a command syntax, keep the following in mind:

- You can use the two colons (::) once in the address to represent the longest successive hexadecimal fields of

zeros.

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

As shown in Figure 2.1, the IPv6 network prefix is composed of the left-most bits of the address. As with an IPv4 address, you can specify the IPv6 prefix using the <prefix>/<prefix-length> format, where the following applies:

The <prefix> parameter is specified as 16-bit hexadecimal values separated by a colon.

The <prefix-length> parameter is specified as a decimal value that indicates the left-most bits of the IPv6 address.

The following is an example of an IPv6 prefix:

2001:FF08:49EA:D088::/64

## IPv6 Address Types

As with IPv4 addresses, you can assign multiple IPv6 addresses to a router interface. Table 2.1 presents the three major types of IPv6 addresses that you can assign to a router interface.

A major difference between IPv4 and IPv6 addresses is that IPv6 addresses support ***scope***, which describes the topology in which the address may be used as a unique identifier for an interface or set of interfaces.

Unicast and multicast addresses support scoping as follows:

- Unicast addresses support two types of scope: global scope and local scope. In turn, local scope supports site-local addresses and link-local addresses. Table 2.1 describes global, site-local, and link-local addresses and the topologies in which they are used.

- Multicast addresses support a scope field, which Table 2.1 describes.

**Table 2.1: IPv6 address types**

| Address Type | Description | Address Structure |
|---|---|---|
| Unicast | An address for a single interface. A packet sent to a unicast address is delivered to the interface identified by the address. | Depends on the type of the unicast address:<br>• Aggregatable global address—An address equivalent to a global or public IPv4 address. The address structure is as follows: a fixed prefix of 2000::/3 (001), a 45-bit global routing prefix, a 16-bit subnet ID, and a 64-bit interface ID.<br>• Site-local address—An address used within a site or intranet. (This address is similar to a private IPv4 address.) A site consists of multiple network links. The address structure is as follows: a fixed prefix of FEC0::/10 (1111 1110 11), a 16-bit subnet ID, and a 64-bit interface ID.<br>• Link-local address—An address used between directly connected nodes on a single network link. The address structure is as follows: a fixed prefix of FE80::/10 (1111 1110 10) and a 64-bit interface ID.<br>• IPv4-compatible address—An address used in IPv6 transition mechanisms that tunnel IPv6 packets dynamically over IPv4 infrastructures. The address embeds an IPv4 address in the low-order 32 bits and the high-order 96 bits are zeros. The address structure is as follows: 0:0:0:0:0:0:A.B.C.D.<br>• Loopback address—An address (0:0:0:0:0:0:0:1 or ::1) that a router can use to send an IPv6 packet to itself. You cannot assign a loopback address to a physical interface.<br>• Unspecified address—An address (0:0:0:0:0:0:0:0 or ::) that a node can use until you configure an IPv6 address for it. |
| Multicast | An address for a set of interfaces belonging to different nodes. Sending a packet to a multicast address results in the delivery of the packet to all interfaces in the set. | A multicast address has a fixed prefix of FF00::/8 (1111 1111). The next 4 bits define the address as a permanent or temporary address. The next 4 bits define the scope of the address (node, link, site, organization, global). |
| Anycast | An address for a set of interfaces belonging to different nodes. Sending a packet to an anycast address results in the delivery of the packet to the closest interface identified by the address. | An anycast address looks similar to a unicast address, because it is allocated from the unicast address space. If you assign a unicast address to multiple interfaces, it is an anycast address. An interface assigned an anycast address must be configured to recognize the address as an anycast address.<br>An anycast address can be assigned to a router only.<br>An anycast address must not be used as the source address of an IPv6 packet. |

A router automatically configures a link-local unicast address for an interface by using the prefix of FE80::/10 (1111 1110 10) and a 64-bit interface ID. The 128-bit IPv6 address is then subjected to duplicate address detection to ensure that the address is unique on the link. If desired, you can override this automatically configured

address by explicitly configuring an address. For more information about explicitly configuring this address, see “Configuring IPv6 on Each Router Interface” on page 3-2.

## IPv6 Stateless Autoconfiguration

Foundry routers use the IPv6 stateless autoconfiguration feature to enable a host on a local link to automatically configure its interfaces with new and globally unique IPv6 addresses associated with its location. The automatic configuration of a host interface is performed without the use of a server, such as a Dynamic Host Configuration Protocol (DHCP) server, or manual configuration.

The automatic configuration of a host interface works in the following way: a router on a local link periodically sends router advertisement messages containing network-type information, such as the 64-bit prefix of the local link and the default route, to all nodes on the link. When a host on the link receives the message, it takes the local link prefix from the message and appends a 64-bit interface ID, thereby automatically configuring its interface. (The 64-bit interface ID is derived from the MAC address of the host’s NIC.) The 128-bit IPv6 address is then subjected to duplicate address detection to ensure that the address is unique on the link.

The duplicate address detection feature verifies that a unicast IPv6 address is unique before it is assigned to a host interface by the stateless auto configuration feature. Duplicate address detection uses neighbor solicitation messages to verify that a unicast IPv6 address is unique. For more information about duplicate address detection, see “Setting Neighbor Solicitation Parameters for Duplicate Address Detection” on page 3-14.

---

**NOTE:** For the stateless auto configuration feature to work properly, the advertised prefix length in router advertisement messages must always be 64 bits. For more information about the router advertisement message, see “Router Advertisement and Solicitation Messages” on page 3-13.

---

The IPv6 stateless autoconfiguration feature can also automatically reconfigure a host’s interfaces if you change the ISP for the host’s network. (The host’s interfaces must be renumbered with the IPv6 prefix of the new ISP.)

The renumbering occurs in the following way: a router on a local link periodically sends advertisements updated with the prefix of the new ISP to all nodes on the link. (The advertisements still contain the prefix of the old ISP.) A host can use the addresses created from the new prefix and the existing addresses created from the old prefix on the link. When you are ready for the host to use the new addresses only, you can configure the lifetime parameters appropriately using the **ipv6 nd prefix-advertisement** command. During this transition, the old prefix is removed from the router advertisements. At this point, only addresses that contain the new prefix are used on the link. For more information about configuring the lifetime parameters, see “Controlling Prefixes Advertised in IPv6 Router Advertisement Messages” on page 3-15.

# Chapter 3
# Configuring Basic IPv6 Connectivity

This chapter explains how to configure basic IPv6 connectivity for a Foundry Layer 3 Switch that supports IPv6. The following mandatory tasks are described:

- Enable IPv6 routing globally on the Foundry Layer 3 Switch
- Configure an IPv6 address or explicitly enable IPv6 on each router interface over which you plan to forward IPv6 traffic
- Configure IPv4 and IPv6 protocol stacks (This step is mandatory only if you want a router interface to send and receive both IPv4 and IPv6 traffic)

The following optional configuration tasks are also described:

- Configure IPv6 Domain Name Server (DNS) resolver
- Configure ECMP Load Sharing for IPv6
- Configure IPv6 ICMP
- Configure the IPv6 neighbor discovery feature
- Change the IPv6 MTU
- Configure an unnumbered interface
- Configure static neighbor entries
- Limit the hop count of an IPv6 packet
- Configure Quality of Service (QoS) for IPv6 traffic

## Enabling IPv6 Routing

By default, IPv6 routing is disabled. To enable the forwarding of IPv6 traffic globally on the router, enter the following command:

```
BigIron(config)# ipv6 unicast-routing
```

***Syntax:*** [no] ipv6 unicast-routing

To disable the forwarding of IPv6 traffic globally on the Foundry device, enter the **no** form of this command.

# Configuring IPv6 on Each Router Interface

To forward IPv6 traffic on a router interface, the interface must have an IPv6 address, or IPv6 must be explicitly enabled. By default, an IPv6 address is not configured on a router interface.

If you choose to configure a global or site-local IPv6 address for an interface, IPv6 is also be enabled on the interface. In addition, when you configure a global or site-local IPv6 address, you must decide on one of the following in the low-order 64 bits:

- A manually configured interface ID
- An automatically computed EUI-64 interface ID

If you prefer to assign a link-local IPv6 address to the interface, you must explicitly enable IPv6 on the interface, which causes a link-local address to be automatically computed for the interface. If preferred, you can override the automatically configured link-local address with an address that you manually configure.

This section provides the following information:

- Configuring a global or site-local address with a manually configured or automatically computed interface ID for an interface
- Automatically or manually configuring a link-local address for an interface
- Configuring IPv6 anycast addresses

## Configuring a Global or Site-Local IPv6 Address

Configuring a global or site-local IPv6 address on an interface does the following:

- Automatically configures an interface ID (a link-local address), if specified
- Enables IPv6 on that interface

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

- Solicited-node multicast group FF02:0:0:0:0:1:FF00::/104 for each unicast address assigned to the interface.
- All-nodes link-local multicast group FF02::1
- All-routers link-local multicast group FF02::2

The neighbor discovery feature sends messages to these multicast groups. For more information, see “Configuring IPv6 Neighbor Discovery” on page 3-12.

### Configuring a Global or Site-Local IPv6 Address with a Manually Configured Interface ID

To configure a global or site-local IPv6 address, including a manually configured interface ID, for an interface, enter commands such as the following:

```
BigIron(config)# interface ethernet 3/1
BigIron(config-if-e100-3/1)# ipv6 address 2001:200:12D:1300:240:D0FF:
FE48:4672:/64
```

These commands configure the global prefix 2001:200:12d:1300::/64 and the interface ID ::240:D0FF:FE48:4672, and enable IPv6 on Ethernet interface 3/1.

***Syntax:*** ipv6 address <ipv6-prefix>/<prefix-length>

You must specify the <ipv6-prefix> parameter in hexadecimal using 16-bit values between colons as documented in RFC 2373.

You must specify the <prefix-length> parameter as a decimal value. A slash mark (/) must follow the <ipv6-prefix> parameter and precede the <prefix-length> parameter.

### Configuring a Global or Site-Local IPv6 Address with an Automatically Computed EUI-64 Interface ID

To configure a global or site-local IPv6 address with an automatically computed EUI-64 interface ID in the low-order 64-bits, enter commands such as the following:

```
BigIron(config)# interface ethernet 3/1
BigIron(config-if-e100-3/1)# ipv6 address 2001:200:12D:1300::/64 eui-64
```

These commands configure the global prefix 2001:200:12d:1300::/64 and an interface ID, and enable IPv6 on Ethernet interface 3/1.

***Syntax:*** ipv6 address <ipv6-prefix>/<prefix-length> eui-64

You must specify the <ipv6-prefix> parameter in hexadecimal using 16-bit values between colons as documented in RFC 2373.

You must specify the <prefix-length> parameter as a decimal value. A slash mark (/) must follow the <ipv6-prefix> parameter and precede the <prefix-length> parameter.

The **eui-64** keyword configures the global or site-local address with an EUI-64 interface ID in the low-order 64 bits. The interface ID is automatically constructed in IEEE EUI-64 format using the interface's MAC address.

## Configuring a Link-Local IPv6 Address

To explicitly enable IPv6 on a router interface without configuring a global or site-local address for the interface, enter commands such as the following:

```
BigIron(config)# interface ethernet 3/1
BigIron(config-if-e100-3/1)# ipv6 enable
```

These commands enable IPv6 on Ethernet interface 3/1 and specify that the interface is assigned an automatically computed link-local address.

***Syntax:*** [no] ipv6 enable

---

**NOTE:** When configuring VLANs that share a common tagged interface with a Virtual Ethernet (VE) interface, Foundry recommends that you override the automatically computed link-local address with a manually configured unique address for the interface. If the interface uses the automatically computed address, which in the case of VE interfaces is derived from a global MAC address, all VE interfaces will have the same MAC address.

---

To override a link-local address that is automatically computed for an interface with a manually configured address, enter commands such as the following:

```
BigIron(config)# interface ethernet 3/1
BigIron(config-if-e100-3/1)# ipv6 address FE80::240:D0FF:FE48:4672 link-local
```

These commands explicitly configure link-local address FE80::240:D0FF:FE48:4672 for Ethernet interface 3/1.

***Syntax:*** ipv6 address <ipv6-address> link-local

You must specify the <ipv6-address> parameter in hexadecimal using 16-bit values between colons as documented in RFC 2373.

The **link-local** keyword indicates that the router interface should use the manually configured link-local address instead of the automatically computed link-local address.

## Configuring IPv6 Anycast Addresses

In IPv6, an ***anycast*** address is an address for a set of interfaces belonging to different nodes. Sending a packet to an anycast address results in the delivery of the packet to the closest interface configured with the anycast address.

An anycast address looks similar to a unicast address, because it is allocated from the unicast address space. If you assign an IPv6 unicast address to multiple interfaces, it is an anycast address. On the Foundry device, you configure an interface assigned an anycast address to recognize the address as an anycast address.

For example, the following commands configure an anycast address on interface 2/1:

```
BigIron(config)# int e 2/1
BigIron(config-if-e100-2/1)# ipv6 address 2002::6/64 anycast
```

***Syntax:*** ipv6 address <ipv6-prefix>/<prefix-length> [anycast]

IPv6 anycast addresses are described in detail in RFC 1884. See RFC 2461 for a description of how the IPv6 Neighbor Discovery mechanism handles anycast addresses.

## Configuring the Management Port for an IPv6 Automatic Address Configuration

With Terathon IronWare release 02.1.00, a BigIron MG8 and a NetIron 40G can have the management port configured to automatically obtain an IPv6 address. This process is the same for any other port and is described in detail in the “Configuring a Global or Site-Local IPv6 Address with an Automatically Computed EUI-64 Interface ID” on page 3-3

## Configuring an IPv6 Host Address for a BigIron MG8 Running a Switch Image

**NOTE:** This feature is only available on the BigIron MG8 when it is configured as a switch. For this feature to work it must have the CHD code enabled on the BigIron MG8.

In the router configuration, each port can be configured separately with an IPv6 address. This is accomplished using the interface configuration process that is described in the "Configuring IPv6 on Each Router Interface" section of the *Foundry IPv6 Configuration Guide*.

When a BigIron MG8 is running a switch-only image of the code, individual ports cannot be configured with an IP address (IPv4 or IPv6). In this situation theBigIron MG8 has one IP address for the management port, and one IP address for the system. This has previously been supported for IPv4 but not IPv6.

In Terathon IronWare release 02.1.00 and later, there is support for configuring an IPv6 address on the management port as described in “Configuring the Management Port for an IPv6 Automatic Address Configuration” on page 3-4 and for configuring a system-wide IPv6 address on the BigIron MG8 in switch mode. Configuration of the system-wide IPv6 address is exactly like configuration of an IPv6 address in router mode except that all of the IPv6 configuration is at the Global Config level instead of at the Interface Config level.

The process for defining the system-wide interface for IPv6 is described in the following sections:

- “Configuring a Global or Site-Local IPv6 Address with a Manually Configured Interface ID as the Switch’s System-wide Address” on page 3-5
- “Configuring a Global or Site-Local IPv6 Address with an Automatically Computed EUI-64 Interface ID as the Switch’s System-wide Address” on page 3-5
- “Configuring a Link-Local IPv6 Address as the Switch’s System-Wide Address” on page 3-5

## Configuring a Global or Site-Local IPv6 Address with a Manually Configured Interface ID as the Switch's System-wide Address

To configure a global or site-local IPv6 address with a manually configured interface ID as a switch's system-wide address, enter a command such as the following at the Global Config level:

```
BigIron MG8(config)#ipv6 address 2001:200:12D:1300:240:D0FF:FE48:4000:1/64
```

***Syntax:*** ipv6 address <ipv6-prefix>/<prefix-length>

You must specify the <ipv6-prefix> parameter in hexadecimal using 16-bit values between colons as documented in RFC 2373.

You must specify the <prefix-length> parameter in decimal value. A slash mark (/) must follow the <ipv6-prefix> parameter and precede the <prefix-length> parameter.

## Configuring a Global or Site-Local IPv6 Address with an Automatically Computed EUI-64 Interface ID as the Switch's System-wide Address

To configure a global or site-local IPv6 address with an automatically computed EUI-64 interface ID in the low order 64-bits s the system-wide address, enter commands such as the following:

```
BigIron(config)# ipv6 address 2001:200:12D:1300::/64 eui-64
```

These commands configure the global prefix 2001:200:12d:1300::/64 and an interface ID as the system-wide address, and enable IPv6.

***Syntax:*** ipv6 address <ipv6-prefix>/<prefix-length> eui-64

You must specify the <ipv6-prefix> parameter in hexadecimal using 16-bit values between colons as documented in RFC 2373.

You must specify the <prefix-length> parameter as a decimal value. A slash mark (/) must follow the <ipv6-prefix> parameter and precede the <prefix-length> parameter.

The **eui-64** keyword configures the global or site-local address with an EUI-64 interface ID in the low-order 64 bits. The interface ID is automatically constructed in IEEE EUI-64 format using the interface's MAC address.

## Configuring a Link-Local IPv6 Address as the Switch's System-Wide Address

To enable IPv6 and automatically configure a global interface enter commands such as the following:

```
BigIron MG8(config)# ipv6 enable
```

This command enables IPv6 on the switch and specifies that the interface is assigned an automatically computed link-local address.

***Syntax:*** [no] ipv6 enable

To override a link-local address that is automatically computed for the global interface with a manually configured address, enter a command such as the following:

```
BigIron(config)# ipv6 address FE80::240:D0FF:FE48:4672 link-local
```

This command explicitly configures the link-local address FE80::240:D0FF:FE48:4672 for the global interface.

***Syntax:*** ipv6 address <ipv6-address> link-local

You must specify the <ipv6-address> parameter in hexadecimal using 16-bit values between colons as documented in RFC 2373.

The **link-local** keyword indicates that the router interface should use the manually configured link-local address instead of the automatically computed link-local address.

# Configuring IPv4 and IPv6 Protocol Stacks

One situation in which you must configure a router to run both IPv4 and IPv6 protocol stacks is if it is deployed as an endpoint for an IPv6 over IPv4 tunnel. For more information, see “IPv6 Over IPv4 Tunnels” on page 9-2.

Each router interface that you want to send and receive both IPv4 and IPv6 traffic must be configured with an IPv4 address and an IPv6 address. (An alternative to configuring a router interface with an IPv6 address is to explicitly enable IPv6 using the **ipv6 enable** command. For more information about using this command, see “Configuring a Link-Local IPv6 Address” on page 3-3.)

To configure a router interface to support both the IPv4 and IPv6 protocol stacks, use commands such as the following:

```
BigIron(config)# ipv6 unicast-routing
BigIron(config)# interface ethernet 3/1
BigIron(config-if-e100-3/1)# ip address 192.168.1.1 255.255.255.0
BigIron(config-if-e100-3/1)# ipv6 address 2001:200:12d:1300::/64 eui-64
```

These commands globally enable IPv6 routing on the router and configure an IPv4 address and an IPv6 address for Ethernet interface 3/1.

***Syntax:*** [no] ipv6 unicast-routing

To disable IPv6 traffic globally on the router, enter the **no** form of this command.

***Syntax:*** ip address <ip-address> <sub-net-mask> [secondary]

You must specify the <ip-address> parameter using 8-bit values in dotted decimal notation.

You can specify the <sub-net-mask> parameter in either dotted decimal notation or as a decimal value preceded by a slash mark (/).

The **secondary** keyword specifies that the configured address is a secondary IPv4 address.

To remove the IPv4 address from the interface, enter the **no** form of this command.

***Syntax:*** ipv6 address <ipv6-prefix>/<prefix-length> [eui-64]

This syntax specifies a global or site-local IPv6 address. For information about configuring a link-local IPv6 address, see “Configuring a Link-Local IPv6 Address” on page 3-3.

You must specify the <ipv6-prefix> parameter in hexadecimal using 16-bit values between colons as documented in RFC 2373.

You must specify the <prefix-length> parameter as a decimal value. A slash mark (/) must follow the <ipv6-prefix> parameter and precede the <prefix-length> parameter.

The **eui-64** keyword configures the global or site-local address with an EUI-64 interface ID in the low-order 64 bits. The interface ID is automatically constructed in IEEE EUI-64 format using the interface’s MAC address. If you do not specify the **eui-64** keyword, you must manually configure the 64-bit interface ID as well as the 64-bit network prefix. For more information about manually configuring an interface ID, see “Configuring a Global or Site-Local IPv6 Address” on page 3-2.

# Configuring IPv6 Domain Name Server (DNS) Resolver

The Domain Name Server (DNS) resolver feature lets you use a host name to perform Telnet, ping, and traceroute commands. You can also define a DNS domain on a Foundry device and thereby recognize all hosts within that domain. After you define a domain name, the Foundry device automatically appends the appropriate domain to the host and forwards it to the domain name server.

For example, if the domain “newyork.com” is defined on a Foundry device, and you want to initiate a ping to host “NYC01” on that domain, you need to reference only the host name in the command instead of the host name and its domain name. For example, you could enter either of the following commands to initiate the ping:

```
BigIron# ping nyc01
BigIron# ping nyc01.newyork.com
```

## Defining a DNS Entry

You can define up to four DNS servers for each DNS entry. The first entry serves as the primary default address. If a query to the primary address fails to be resolved after three attempts, the next gateway address is queried (also up to three times). This process continues for each defined gateway address until the query is resolved. The order in which the default gateway addresses are polled is the same as the order in which you enter them.

Suppose you want to define the domain name of newyork.com on a Foundry device and then define four possible default DNS gateway addresses. To do so using IPv4 addressing, you would enter the following commands:

```
BigIron(config)# ip dns domain-name newyork.com
BigIron(config)# ip dns server-address 209.157.22.199 205.96.7.15 208.95.7.25
201.98.7.15
```

***Syntax:*** ip dns server-address <ip-addr> [<ip-addr>] [<ip-addr>] [<ip-addr>]

In this example, the first IP address in the **ip dns server-address...** command becomes the primary gateway address and all others are secondary addresses. Because IP address 201.98.7.15 is the last address listed, it is also the last address consulted to resolve a query.

**NOTE:** This feature is not supported for BigIron software release 08.0.00.

### Defining an IPv6 DNS Entry

IPv6 defines new DNS record types to resolve queries for domain names to IPv6 addresses, as well as IPv6 addresses to domain names. Foundry devices running IPv6 software support AAAA DNS records, which are defined in RFC 1886.

AAAA DNS records are analogous to the A DNS records used with IPv4. They store a complete IPv6 address in each record. AAAA records have a type value of 28.

To establish an IPv6 DNS entry for the device, enter the following command:

```
BigIron(config)# ipv6 dns domain-name companynet.com
```

***Syntax:*** [no] ipv6 dns domain-name <domain name>

To define an IPv6 DNS server address, enter the following command:

```
BigIron(config)# ipv6 dns server-address 200::1
```

***Syntax:*** [no] ipv6 dns server-address <ipv6-addr> [<ipv6-addr>] [<ipv6-addr>] [<ipv6-addr>]

As an example, in a configuration where ftp6.companynet.com is a server with an IPv6 protocol stack, when a user pings ftp6.companynet.com, the Foundry device attempts to resolve the AAAA DNS record. In addition, if the DNS server does not have an IPv6 address, as long as it is able to resolve AAAA records, it can still respond to DNS queries.

# ECMP Load Sharing for IPv6

The IPv6 route table selects the best route to a given destination from among the routes in the tables maintained by the configured routing protocols (BGP4, OSPF, static, and so on). The IPv6 route table can contain more than one path to a given destination. When this occurs, the Foundry device selects the path with the lowest cost for insertion into the routing table. If more than one path with the lowest cost exists, all of these paths are inserted into the routing table, subject to the configured maximum number of load sharing paths (by default 4). The device uses ***Equal-Cost Multi-Path (ECMP) load sharing*** to select a path to a destination.

When the device receives traffic for a destination, and the IPv6 route table contains multiple, equal-cost paths to that destination, the device checks the ***IPv6 forwarding cache*** for a forwarding entry for the destination. The IPv6 forwarding cache provides a fast path for forwarding IPv6 traffic. The IPv6 forwarding cache contains entries that associate a destination host or network with a path (next-hop router).

If the IPv6 forwarding cache contains a forwarding entry for the destination, the Foundry device uses the entry to forward the traffic. If the IPv6 forwarding cache does not contain a forwarding entry for the destination, the software selects a path from among the available equal-cost paths to the destination, then creates an entry in the in the cache based on the calculation. Subsequent traffic for the same destination uses the forwarding entry. Entries remain in the IPv6 forwarding cache for one minute, then are aged out.

If the path selected by the device becomes unavailable, its entry in the IPv6 forwarding cache is removed, a new path is selected from the remaining equal-cost paths to the destination, and an entry is created in the IPv6 forwarding cache using the new path.

Foundry devices support the following ECMP load-sharing methods for IPv6 traffic:

- Network-based – The Foundry device distributes traffic across equal-cost paths based on destination network address. The software selects a path based on a calculation involving the maximum number of load-sharing paths allowed and the actual number of paths to the destination network. This is the default ECMP load-sharing method for IPv6.
- Host-based – The Foundry device uses a simple round-robin mechanism to distribute traffic across the equal-cost paths based on destination host IP address. The device uses this ECMP load-sharing method for IPv6 if you explicitly configure it to do so.

You can manually disable or enable ECMP load sharing for IPv6, specify the number of equal-cost paths the device can distribute traffic across, and configure the device to use the host-based ECMP load-sharing method instead of the network-based method. In addition, you can display information about the status of ECMP load-sharing on the device, as well as the entries in the IPv6 forwarding cache.

## Disabling or Re-Enabling ECMP Load Sharing for IPv6

ECMP load sharing for IPv6 is enabled by default. To disable the feature, enter the following command:

```
BigIron(config)# no ipv6 load-sharing
```

If you want to re-enable the feature after disabling it, enter the following command:

```
BigIron(config)# ipv6 load-sharing
```

***Syntax:*** [no] ipv6 load-sharing

## Changing the Maximum Number of Load Sharing Paths for IPv6

By default, IPv6 ECMP load sharing allows traffic to be balanced across up to four equal paths. You can change the maximum number of paths the device supports to a value from 2 – 8.

To change the number of ECMP load sharing paths for IPv6, enter a command such as the following:

```
BigIron(config)# ipv6 load-sharing 8
```

***Syntax:*** [no] ipv6 load-sharing [<num>]

The <num> parameter specifies the number of paths and can be from 2 – 8. The default is 4.

## Changing the ECMP Load-Sharing Method for IPv6

Foundry devices can perform ECMP load-sharing for IPv6 traffic based on destination host address or destination network. The default is network-based IP load sharing. If you want to enable the device to perform host-based IP load sharing instead, enter the following command:

```
BigIron(config)# ipv6 load-sharing by-host
```

***Syntax:*** [no] ipv6 load-sharing by-host

This command enables host-based ECMP load sharing on the device. The command also disables network-based ECMP load-sharing at the same time.

## DHCP Relay Agent for IPv6

A client locates a DHCP server using a reserved, link-scoped multicast address. For this reason, it is a requirement for direct communication between the client and the server that they be attached by the same link. However, in some situations in which ease of management, economy, and scalability is a concern, it is useful to allow a DHCP client to send a message to a DHCP server by using a DHCP relay agent. A DHCP relay agent, which may reside on the clients link, is used to relay messages between the client and the server. A DHCP relay agent is transparent to the client.

When the relay agent receives a message to be relayed from a client to another relay agent or a DHCP server, it creates a new Relay-forward message, puts the original DHCP message to relay forward option, and includes its own address and the address it received is in the same option.

### Configuring DHCP for IPv6 Relay Agent

You can enable the DHCP for IPv6 relay agent function and specify the relay destination addresses on an interface by entering the command at the interface level:

```
BigIron(config)# interface ethernet 2/3
```

```
BigIron(config-if-e10000-2/3)#ipv6 dhcp-relay-dest FE80::250:A2FF:FEBF:A056
ethernet 1/3
```

***Syntax:*** ipv6 dhcp-relay *< ipv6-address >*

Select the **ipv6-address** to specify a destination address to which the client messages are forwarded and enables DHCP for IPv6 relay services on the interface.

## Enabling Support for Network-Based ECMP Load Sharing for IPv6 (BigIron MG8 and NetIron 40G Software Release 02.1.00)

In previous releases of BigIron MG8 and NetIron 40G software, only ECMP Load sharing by host was supported for IPv6. In that configuration, a simple round-robin mechanism is employed to distribute traffic across equal-cost paths based on the destination host IP address. Routes to each destination host are stored in CAM and accessed when a path to a host is required.

With release 02.1.00, network-based ECMP load sharing is also supported. If this configuration is selected, traffic is distributed across equal-cost paths based on the destination network address. Routes to each network are stored in CAM and accessed when a path to a network is required. Because multiple hosts are likely to reside on a network, this method uses fewer CAM entries than load sharing by host. When you select network-based ECMP load sharing, you can choose either of the following two CAM modes:

- **Dynamic Mode** – In the dynamic mode, routes are entered into the CAM dynamically using a flow-based scheme. In this mode routes are only added to the CAM as they are required. Once routes are added to the CAM, they are subject to being aged-out when they are not in use. Because this mode conserves CAM, it is useful for situations where CAM resources are stressed or limited.
- **Static Mode** – In the static mode, routes are entered into the CAM whenever they are discovered. Routes aren't aged once routes are added to the CAM and they are subject to being aged-out when they are not in use.

### Configuring the CAM Mode to Support Network-based ECMP Load Sharing for IPv6

To configure the CAM mode to support network-based ECMP load sharing for IPv6, use a command such as the following at the Global Configuration level:

```
BigIron MG8(config)# #cam-mode ipv6 dynamic
```

***Syntax:*** [no] cam-mode ipv6 [dynamic | static | host ]

The **dynamic** parameter configures the BigIron MG8 and NetIron 40G for network-based ECMP load sharing using the dynamic CAM mode.

The **static** parameter configures the BigIron MG8 and NetIron 40G for network-based ECMP load sharing using the static CAM mode.

The **host** parameter configures the BigIron MG8 and NetIron 40G for host-based ECMP load sharing using the dynamic CAM mode.

You must reload the router for this command to take effect.

## Displaying ECMP Load-Sharing Information for IPv6

To display the status of ECMP load sharing for IPv6, enter the following command:

```
BigIron# show ipv6
Global Settings

  unicast-routing enabled, hop-limit 64
  No Inbound Access List Set
  No Outbound Access List Set
  Prefix-based IPv6 Load-sharing is Enabled, Number of load share paths: 4
```

***Syntax:*** show ipv6

You can display the entries in the IPv6 forwarding cache; for example

:

```
BigIron# show ipv6 cache
Total number of cache entries: 10
    IPv6 Address                          Next Hop                       Port
1   5000:2::2                             LOCAL                          tunnel 2
2   2000:4::106                           LOCAL                          ethe 2
3   2000:4::110                           DIRECT                         ethe 2
4   2002:c0a8:46a::1                      LOCAL                          ethe 2
5   fe80::2e0:52ff:fe99:9737              LOCAL                          ethe 2
6   fe80::ffff:ffff:feff:ffff             LOCAL                          loopback 2
7   fe80::c0a8:46a                        LOCAL                          tunnel 2
8   fe80::c0a8:46a                        LOCAL                          tunnel 6
9   2999::1                               LOCAL                          loopback 2
10  fe80::2e0:52ff:fe99:9700              LOCAL                          ethe 1
```

***Syntax:*** show ipv6 cache [<index-number> | <ipv6-prefix>/<prefix-length> | <ipv6-address> | ethernet <port> | ve <number> | tunnel <number>]

# Configuring IPv6 ICMP

As with the Internet Control Message Protocol (ICMP) for IPv4, ICMP for IPv6 provides error and informational messages. Foundry's implementation of the stateless autoconfiguration, neighbor discovery, and path MTU discovery features use ICMP messages.

This section explains how to configure the following IPv6 ICMP features:

- ICMP rate limiting.
- ICMP redirects.

## Configuring ICMP Rate Limiting

You can limit the rate at which IPv6 ICMP error messages are sent out on a network. IPv6 ICMP implements a token bucket algorithm.

To illustrate how this algorithm works, imagine a virtual bucket that contains a number of tokens. Each token represents the ability to send one ICMP error message. Tokens are placed in the bucket at a specified interval until the maximum number of tokens allowed in the bucket is reached. For each error message that ICMP sends, a token is removed from the bucket. If ICMP generates a series of error messages, messages can be sent until the bucket is empty. If the bucket is empty of tokens, error messages cannot be sent until a new token is placed in the bucket.

You can adjust the following elements related to the token bucket algorithm:

- The interval at which tokens are added to the bucket. The default is 100 milliseconds.
- The maximum number of tokens in the bucket. The default is 10 tokens.

For example, to adjust the interval to 1000 milliseconds and the number of tokens to 100 tokens, enter the following command:

```
BigIron(config)# ipv6 icmp error-interval 1000 100
```

***Syntax:*** ipv6 icmp error-interval <interval> [<number-of-tokens>]

The interval in milliseconds at which tokens are placed in the bucket can range from 0 – 2147483647. The maximum number of tokens stored in the bucket can range from 1 – 200.

**NOTE:** If you retain the default interval value or explicitly set the value to 100 milliseconds, output from the **show run** command does not include the setting of the **ipv6 icmp error-interval** command because the setting is the default.

Also, if you configure the interval value to a number that does not evenly divide into 100000 (100 milliseconds), the system rounds up the value to a next higher value that does divide evenly into 100000. For example, if you specify an interval value of 150, the system rounds up the value to 200.

ICMP rate limiting is enabled by default. To disable ICMP rate limiting, set the interval to zero.

### Disabling or Reenabling ICMP Redirect Messages

You can disable or re-enable the sending of ICMP redirect messages by a router. By default, a router can send an ICMP redirect message to a neighboring host to inform it of a better first-hop router on a path to a destination. No further configuration is required to enable the sending of ICMP redirect messages. (For more information about how ICMP redirect messages are implemented for IPv6, see “Configuring IPv6 Neighbor Discovery” on page 3-12.)

For example, to disable the sending of ICMP redirect messages on Ethernet interface 3/1, enter the following commands:

```
BigIron(config)# interface ethernet 3/1
BigIron(config-if-e100-3/1)# no ipv6 redirects
```

***Syntax:*** [no] ipv6 redirects

To reenable the sending of ICMP redirect messages on Ethernet interface 3/1, enter the following commands:

```
BigIron(config)# interface ethernet 3/1
BigIron(config-if-e100-3/1)# ipv6 redirects
```

Use the **show ipv6 interface** <interface> <port-number> command to verify that the sending of ICMP redirect messages is enabled on a particular interface.

## Configuring IPv6 Neighbor Discovery

The neighbor discovery feature for IPv6 uses IPv6 ICMP messages to do the following:

- Determine the link-layer address of a neighbor on the same link.
- Verify that a neighbor is reachable.
- Track neighbor routers.

An IPv6 host is required to listen for and recognize the following addresses that identify itself:

- Link-local address.
- Assigned unicast address.
- Loopback address.
- All-nodes multicast address.
- Solicited-node multicast address.
- Multicast address to all other groups to which it belongs.

You can adjust the following IPv6 neighbor discovery features:

- Neighbor solicitation messages for duplicate address detection.
- Router advertisement messages:
  - Interval between router advertisement messages.
  - Value that indicates a router is advertised as a default router (for use by all nodes on a given link).

  - Prefixes advertised in router advertisement messages.
  - Flags for host stateful autoconfiguration.
- Amount of time during which an IPv6 node considers a remote node reachable (for use by all nodes on a given link).

## Neighbor Solicitation and Advertisement Messages

Neighbor solicitation and advertisement messages enable a node to determine the link-layer address of another node (neighbor) on the same link. (This function is similar to the function provided by the Address Resolution Protocol [ARP] in IPv4.) For example, node 1 on a link wants to determine the link-layer address of node 2 on the same link. To do so, node 1, the source node, multicasts a neighbor solicitation message. The neighbor solicitation message, which has a value of 135 in the Type field of the ICMP packet header, contains the following information:

- Source address: IPv6 address of node 1 interface that sends the message.
- Destination address: solicited-node multicast address (FF02:0:0:0:0:1:FF00::/104) that corresponds the IPv6 address of node 2.
- Link-layer address of node 1.
- A query for the link-layer address of node 2.

After receiving the neighbor solicitation message from node 1, node 2 replies by sending a neighbor advertisement message, which has a value of 136 in the Type field of the ICMP packet header. The neighbor solicitation message contains the following information:

- Source address: IPv6 address of the node 2 interface that sends the message.
- Destination address: IPv6 address of node 1.
- Link-layer address of node 2.

After node 1 receives the neighbor advertisement message from node 2, nodes 1 and 2 can now exchange packets on the link.

After the link-layer address of node 2 is determined, node 1 can send neighbor solicitation messages to node 2 to verify that it is reachable. Also, nodes 1, 2, or any other node on the same link can send a neighbor advertisement message to the all-nodes multicast address (FF02::1) if there is a change in their link-layer address.

## Router Advertisement and Solicitation Messages

Router advertisement and solicitation messages enable a node on a link to discover the routers on the same link.

Each configured router interface on a link sends out a router advertisement message, which has a value of 134 in the Type field of the ICMP packet header, periodically to the all-nodes link-local multicast address (FF02::1).

A configured router interface can also send a router advertisement message in response to a router solicitation message from a node on the same link. This message is sent to the unicast IPv6 address of the node that sent the router solicitation message.

At system startup, a host on a link sends a router solicitation message to the all-routers multicast address (FF01). Sending a router solicitation message, which has a value of 133 in the Type field of the ICMP packet header, enables the host to automatically configure its IPv6 address immediately instead of awaiting the next periodic router advertisement message.

Because a host at system startup typically does not have a unicast IPv6 address, the source address in the router solicitation message is usually the unspecified IPv6 address (0:0:0:0:0:0:0:0). If the host has a unicast IPv6 address, the source address is the unicast IPv6 address of the host interface sending the router solicitation message.

Entering the **ipv6 unicast-routing** command automatically enables the sending of router advertisement messages on all configured router Ethernet interfaces. You can configure several router advertisement message parameters. For information about disabling the sending of router advertisement messages and the router advertisement parameters that you can configure, see “Enabling and Disabling IPv6 Router Advertisements” on page 3-16 and “Setting IPv6 Router Advertisement Parameters” on page 3-14.

## Neighbor Redirect Messages

After forwarding a packet, by default, a router can send a neighbor redirect message to a host to inform it of a better first-hop router. The host receiving the neighbor redirect message will then readdress the packet to the better router.

A router sends a neighbor redirect message only for unicast packets, only to the originating node, and to be processed by the node.

A neighbor redirect message has a value of 137 in the Type field of the ICMP packet header.

## Setting Neighbor Solicitation Parameters for Duplicate Address Detection

Although the stateless autoconfiguration feature assigns the 64-bit interface ID portion of an IPv6 address using the MAC address of the host's NIC, duplicate MAC addresses can occur. Therefore, the duplicate address detection feature verifies that a unicast IPv6 address is unique before it is assigned to a host interface by the stateless autoconfiguration feature. Duplicate address detection verifies that a unicast IPv6 address is unique.

If duplicate address detection identifies a duplicate unicast IPv6 address, the address is not used. If the duplicate address is the link-local address of the host interface, the interface stops processing IPv6 packets.

You can configure the following neighbor solicitation message parameters that affect duplicate address detection while it verifies that a tentative unicast IPv6 address is unique:

- The number of consecutive neighbor solicitation messages that duplicate address detection sends on an interface. By default, duplicate address detection sends three neighbor solicitation messages without any follow-up messages.
- The interval in seconds at which duplicate address detection sends a neighbor solicitation message on an interface. By default, duplicate address detection sends a neighbor solicitation message every 1 second.

**NOTE:** For the interval at which duplicate address detection sends a neighbor solicitation message on an interface, the Foundry device uses seconds as the unit of measure instead of milliseconds.

For example, to change the number of neighbor solicitation messages sent on Ethernet interface 3/1 to two and the interval between the transmission of the two messages to 9 seconds, enter the following commands:

```
BigIron(config)# interface ethernet 3/1
BigIron(config-if-e100-3/1)# ipv6 nd dad attempt 2
BigIron(config-if-e100-3/1)# ipv6 nd ns-interval 9
```

***Syntax:*** [no] ipv6 nd dad attempt <number>

***Syntax:*** [no] ipv6 nd ns-interval <number>

For the number of neighbor solicitation messages, you can specify any number of attempts. Configuring a value of 0 disables duplicate address detection processing on the specified interface. To restore the number of messages to the default value, use the **no** form of this command.

For the interval between neighbor solicitation messages, you can specify any number of seconds. Foundry does not recommend very short intervals in normal IPv6 operation. When a nondefault value is configured, the configured time is both advertised and used by the router itself. To restore the default interval, use the **no** form of this command.

## Setting IPv6 Router Advertisement Parameters

You can adjust the following parameters for router advertisement messages:

- The interval (in seconds) at which an interface sends router advertisement messages. By default, an interface sends a router advertisement message every 200 seconds.
- The "router lifetime" value, which is included in router advertisements sent from a particular interface. The value (in seconds) indicates if the router is advertised as a default router on this interface. If you set the value of this parameter to 0, the router is not advertised as a default router on an interface. If you set this parameter to a value that is not 0, the router is advertised as a default router on this interface. By default, the router

lifetime value included in router advertisement messages sent from an interface is 1800 seconds.

When adjusting these parameter settings, Foundry recommends that the interval between router advertisement transmission be less than or equal to the router lifetime value if the router is advertised as a default router. For example, to adjust the interval of router advertisements to 300 seconds and the router lifetime value to 1900 seconds on Ethernet interface 3/1, enter the following commands:

```
BigIron(config)# interface ethernet 3/1
BigIron(config-if-e100-3/1)# ipv6 nd ra-interval 300
BigIron(config-if-e100-3/1)# ipv6 nd ra-lifetime 1900
```

***Syntax:*** [no] ipv6 nd ra-interval <number>

***Syntax:*** [no] ipv6 nd ra-lifetime <number>

The <number> parameter in both commands indicates any numerical value. To restore the default interval or router lifetime value, use the **no** form of the respective command.

## Controlling Prefixes Advertised in IPv6 Router Advertisement Messages

By default, router advertisement messages include prefixes configured as addresses on router interfaces using the **ipv6 address** command. You can use the **ipv6 nd prefix-advertisement** command to control exactly which prefixes are included in router advertisement messages. Along with which prefixes the router advertisement messages contain, you can also specify the following parameters:

- Valid lifetime—(Mandatory) The time interval (in seconds) in which the specified prefix is advertised as valid. The default is 2592000 seconds (30 days). When the timer expires, the prefix is no longer considered to be valid.
- Preferred lifetime—(Mandatory) The time interval (in seconds) in which the specified prefix is advertised as preferred. The default is 604800 seconds (7 days). When the timer expires, the prefix is no longer considered to be preferred.
- Onlink flag—(Optional) If this flag is set, the specified prefix is assigned to the link upon which it is advertised. Nodes sending traffic to addresses that contain the specified prefix consider the destination to be reachable on the local link.
- Autoconfiguration flag—(Optional) If this flag is set, the stateless auto configuration feature can use the specified prefix in the automatic configuration of 128-bit IPv6 addresses for hosts on the local link. For more information, see “IPv6 Stateless Autoconfiguration” on page 2-4.

For example, to advertise the prefix 2001:e077:a487:7365::/64 in router advertisement messages sent out on Ethernet interface 3/1 with a valid lifetime of 1000 seconds, a preferred lifetime of 800 seconds, and the Onlink and Autoconfig flags set, enter the following commands:

```
BigIron(config)# interface ethernet 3/1
BigIron(config-if-e100-3/1)# ipv6 nd prefix-advertisement 2001:e077:a487:7365::/64
1000 800 onlink autoconfig
```

***Syntax:*** [no] ipv6 nd prefix-advertisement <ipv6-prefix>/<prefix-length> <valid-lifetime> <preferred-lifetime> [autoconfig] [onlink]

You must specify the <ipv6-prefix> parameter in hexadecimal using 16-bit values between colons as documented in RFC 2373.

You must specify the <prefix-length> parameter as a decimal value. A slash mark (/) must follow the <ipv6-prefix> parameter and precede the <prefix-length> parameter.

The valid lifetime and preferred lifetime is a numerical value between 0 – 4294967295 seconds. The default valid lifetime is 2592000 seconds (30 days), while the default preferred lifetime is 604800 seconds (7 days).

To remove a prefix from the router advertisement messages sent from a particular interface, use the **no** form of this command.

## Setting Flags in IPv6 Router Advertisement Messages

An IPv6 router advertisement message can include the following flags:

- Managed Address Configuration—This flag indicates to hosts on a local link if they should use the stateful autoconfiguration feature to get IPv6 addresses for their interfaces. If the flag is set, the hosts use stateful autoconfiguration to get addresses as well as non-IPv6-address information. If the flag is not set, the hosts do not use stateful autoconfiguration to get addresses and if the hosts can get non-IPv6-address information from stateful autoconfiguration is determined by the setting of the Other Stateful Configuration flag.
- Other Stateful Configuration—This flag indicates to hosts on a local link if they can get non-IPv6 address autoconfiguration information. If the flag is set, the hosts can use stateful autoconfiguration to get non-IPv6-address information.

**NOTE:** When determining if hosts can use stateful autoconfiguration to get non-IPv6-address information, a set Managed Address Configuration flag overrides an unset Other Stateful Configuration flag. In this situation, the hosts can obtain nonaddress information. However, if the Managed Address Configuration flag is not set and the Other Stateful Configuration flag is set, then the setting of the Other Stateful Configuration flag is used.

By default, the Managed Address Configuration and Other Stateful Configuration flags are not set in router advertisement messages. For example, to set these flags in router advertisement messages sent from Ethernet interface 3/1, enter the following commands:

```
BigIron(config)# interface ethernet 3/1
BigIron(config-if-e100-3/1)# ipv6 nd managed-config-flag
BigIron(config-if-e100-3/1)# ipv6 nd other-config-flag
```

***Syntax:*** [no] ipv6 nd managed-config-flag

***Syntax:*** [no] ipv6 nd other-config-flag

To remove either flag from router advertisement messages sent on an interface, use the **no** form of the respective command.

## Enabling and Disabling IPv6 Router Advertisements

If IPv6 unicast routing is enabled on an Ethernet interface, by default, this interface sends IPv6 router advertisement messages. However, by default, non-LAN interface types, for example, tunnel interfaces, do not send router advertisement messages.

To disable the sending of router advertisement messages on an Ethernet interface, enter commands such as the following:

```
BigIron(config)# interface ethernet 3/1
BigIron(config-if-e100-3/1)# ipv6 nd suppress-ra
```

To enable the sending of router advertisement messages on a tunnel interface, enter commands such as the following:

```
BigIron(config)# interface tunnel 1
BigIron(config-tnif-1)# no ipv6 nd suppress-ra
```

***Syntax:*** [no] ipv6 nd suppress-ra

## Configuring Reachable Time for Remote IPv6 Nodes

You can configure the duration (in seconds) that a router considers a remote IPv6 node reachable. By default, a router interface uses the value of 30 seconds.

The router advertisement messages sent by a router interface include the amount of time specified by the **ipv6 nd reachable-time** command so that nodes on a link use the same reachable time duration. By default, the messages include a default value of 0.

**NOTE:** For the interval at which a router interface sends router advertisement messages, Foundry uses seconds as the unit of measure instead of milliseconds.

Foundry does not recommend configuring a short reachable time duration, because a short duration causes the IPv6 network devices to process the information at a greater frequency.

For example, to configure the reachable time of 40 seconds for Ethernet interface 3/1, enter the following commands:

```
BigIron(config)# interface ethernet 3/1
BigIron(config-if-e100-3/1)# ipv6 nd reachable-time 40
```

***Syntax:*** [no] ipv6 nd reachable-time <seconds>

For the <seconds> parameter, you can specify any numerical value. To restore the default time, use the **no** form of this command.

## Changing the IPv6 MTU

The IPv6 MTU is the maximum length of an IPv6 packet that can be transmitted on a particular interface. If an IPv6 packet is longer than an MTU, the host that originated the packet fragments the packet and transmits its contents in multiple packets that are shorter than the configured MTU. You can configure the MTU on individual interfaces. Per RFC 2460, the minimum IPv6 MTU for any interface is 1280 bytes.

For example, to configure the MTU on Ethernet interface 3/1 as 1280 bytes, enter the following commands:

```
BigIron(config)# interface ethernet 3/1
BigIron(config-if-e100-3/1)# ipv6 mtu 1280
```

***Syntax:*** [no] ipv6 mtu <bytes>

You can specify between 1280 – 1500 bytes. If a nondefault value is configured for an interface, router advertisements include an MTU option.

On the BigIron MG8 and NetIron 40G running software release 02.2.01 and later, you can configure IPv6 MTU for to be greater than 1500 bytes, although the default remains at 1500 bytes. The value of the MTU you can define depends on the following:

- For a physical port, the maximum value of the MTU is the equal to the maximum frame size of the port minus 18 (Layer 2 MAC header + CRC).
- For a virtual routing interface, the maximum value of the MTU is the maximum frame size configured for the VLAN to which it is associated, minus 18 (Layer 2 MAC header + CRC). If a maximum frame size for a VLAN is not configured, then configure the MTU based on the smallest maximum frame size of all the ports of the VLAN that corresponds to the virtual routing interface, minus 18 (Layer 2 MAC header + CRC).

To define IPv6 MTU globally, enter:

```
BigIron MG8(config)#ipv6 mtu 1300
```

To define IPv6 MTU on an interface, enter:

```
BigIron MG8(config-if-e1000-2/1)#ipv6 mtu
```

***Syntax:*** ipv6 mtu <value>

**NOTE:** If a the size of a jumbo packet received on a port is equal to the maximum frame size – 18 (Layer 2 MAC header + CRC) and if this value is greater than the outgoing port's IPv4/IPv6 MTU, then it will be forwarded in the CPU.

## Configuring an Unnumbered Interface

You can enable IPv6 on a tunnel interface but not assign an IPv6 address to the interface. The unnumbered interface feature is useful when you are connecting two isolated IPv6 domains over an IPv4 infrastructure. In this situation, an IPv6 address on the tunnel interface might not serve a purpose.

For example, to configure tunnel interface 1 as an unnumbered port and specify the global IPv6 address of Ethernet 3/1 as the source address, enter the following commands:

```
BigIron(config)# interface tunnel 1
BigIron(config-tnif-1)# ipv6 unnumbered ethernet 3/1
```

***Syntax:*** ipv6 unnumbered <interface> <number>

The syntax of the **ipv6 unnumbered** command requires that you specify the interface type and number of a physical port. The software uses the global IPv6 address of the specified physical port as the source address for IPv6 packets generated by the unnumbered interface.

IPv6 packets that are originated from an unnumbered interface use the global IPv6 address of the interface specified in the **ipv6 unnumbered** command as the source address for the packets.

The interface you specify with the <interface> and <number> parameters must be enabled (listed as "up" in the **show ipv6 interface** command display).

For more information about configuring tunnels, see “IPv6 Over IPv4 Tunnels” on page 9-2.

## Configuring Static Neighbor Entries

In some special cases, a neighbor cannot be reached using the neighbor discovery feature. In this situation, you can add a static entry to the IPv6 neighbor discovery cache, which causes a neighbor to be reachable at all times without using neighbor discovery. (A static entry in the IPv6 neighbor discovery cache functions like a static ARP entry in IPv4.)

For example, to add a static entry for a neighbor with the IPv6 address 3001:ffe0:2678:47b and link-layer address 0004.6a2b.8641 that is reachable through Ethernet interface 3/1, enter the following command:

```
BigIron(config)# ipv6 neighbor 3001:ffe0:2678:47b ethernet 3/1 0004.6a2b.8641
```

***Syntax:*** [no] ipv6 neighbor <ipv6-address> ethernet <port> | ve <ve-number> [ethernet <port>] <link-layer-address>

The <ipv6-address> parameter specifies the address of the neighbor.

The **ethernet | ve** parameter specifies the interface through which to reach a neighbor. If you specify an Ethernet interface, specify the port number of the Ethernet interface. If you specify a VE, specify the VE number and then the Ethernet port numbers associated with the VE. The link-layer address is a 48-bit hardware address of the neighbor.

If you attempt to add an entry that already exists in the neighbor discovery cache, the software changes the already existing entry to a static entry.

To remove a static IPv6 entry from the IPv6 neighbor discovery cache, use the **no** form of this command.

## Limiting the Number of Hops an IPv6 Packet Can Traverse

By default, the maximum number of hops an IPv6 packet can traverse is 64. You can change this value to between 1 – 255 hops. For example, to change the maximum number of hops to 70, you can enter the following command:

```
BigIron(config)# ipv6 hop-limit 70
```

***Syntax:*** [no] ipv6 hop-limit <number>

The number of hops can be from 1 – 255.

# QoS for IPv6 Traffic

Configuring QoS for IPv6 traffic is generally the same as it is for IPv4 traffic. The QoS policies you configure on the Foundry device apply to both incoming IPv6 and IPv4 traffic. However, using IP ACLs to perform QoS for IPv6 traffic is not supported.

To enable QoS for IPv6 traffic, enter the following commands:

```
NI4802 Router(config)# port-priority
NI4802 Router(config)# write memory
NI4802 Router(config)# end
NI4802 Router# reload
```

***Syntax:*** [no] port-priority

---

**NOTE:** You must save the configuration and reload the software to place the change into effect. This applies whether you are enabling QoS for IPv6 or IPv4 traffic.

---

The **port-priority** command globally enables QoS for IPv6 traffic on all 10/100 and 1 Gigabit interfaces. When QoS is enabled with the **port-priority** command, the device inserts a value in the internal Foundry header based on a combination of the following information:

- 802.1p priority
- Interface priority (if configured)
- VLAN priority (if configured)
- The first two bits in the Type of Service (ToS) header

For more information, see the "Configuring IronClad Quality of Service" chapter of the *Foundry Policy and Filter Configuration Guide*.

After QoS is enabled with the **port-priority** command, you can optionally enable advanced ToS-based QoS on individual interfaces. Enabling advanced ToS-based QoS on an interface allows you to specify the trust level and packet marking used for packets received on that interface. The ***trust level*** determines the type of QoS information the device uses for performing QoS. ***Marking*** is the process of changing the packet's QoS information for the next hop.

To enable advanced ToS-based QoS on an interface, enter commands such as the following:

```
NI4802 Router(config)# int e 1
NI4802 Router(config-if-e100-1)# qos-tos
NI4802 Router(config-if-e100-1)# qos-tos trust ip-prec
NI4802 Router(config-if-e100-1)# qos-tos mark dscp
```

***Syntax:*** [no] qos-tos

***Syntax:*** [no] qos-tos trust cos | ip-prec | dscp

***Syntax:*** [no] qos-tos mark cos | dscp

The commands in this example enable advanced ToS-based QoS on interface 1, set the trust level for an interface to IP Precedence, and configure the device to change the outbound packet's DSCP value to match the results of the device's QoS mapping from the specified trust level.

After you enable ToS-based QoS with the **qos-tos** command, there is no default trust level for IPv6 traffic. You must explicitly configure a trust level for IPv6 traffic. When configuring advanced ToS-based QoS, you must specify a trust level to enable DSCP marking.

Note that when advanced QoS is enabled on an interface, the configured trust level on that incoming interface determines the final priority of the packet.

For more information on configuring advanced ToS-based QoS on an interface, see the "Enhanced QoS" chapter of the *Foundry Policy and Filter Configuration Guide*.

# Clearing Global IPv6 Information

You can clear the following global IPv6 information:

- Entries from the IPv6 cache.
- Entries from the IPv6 neighbor table.
- IPv6 routes from the IPv6 route table.
- IPv6 traffic statistics.
- IPv6 session flows

## Clearing the IPv6 Cache

You can remove all entries from the IPv6 cache or specify an entry based on the following:

- IPv6 prefix.
- IPv6 address.
- Interface type.

For example, to remove entries for IPv6 address 2000:e0ff::1, enter the following command at the Privileged EXEC level or any of the Config levels of the CLI:

```
BigIron# clear ipv6 cache 2000:e0ff::1
```

***Syntax:*** clear ipv6 cache [<ipv6-prefix>/<prefix-length> | <ipv6-address> | ethernet <port> | tunnel <number> | ve <number>]

You must specify the <ipv6-prefix> parameter in hexadecimal using 16-bit values between colons as documented in RFC 2373. You must specify the <prefix-length> parameter as a decimal value. A slash mark (/) must follow the <ipv6-prefix> parameter and precede the <prefix-length> parameter.

You must specify the <ipv6-address> parameter in hexadecimal using 16-bit values between colons as documented in RFC 2373.

The **ethernet | tunnel | ve** parameter specifies the interfaces for which you can remove cache entries. If you specify an Ethernet interface, also specify the port number associated with the interface. If you specify a VE or tunnel interface, also specify the VE or tunnel number, respectively.

## Clearing IPv6 Neighbor Information

You can remove all entries from the IPv6 neighbor table or specify an entry based on the following:

- IPv6 prefix.
- IPv6 address.
- Interface type.

For example, to remove entries for Ethernet interface 3/1, enter the following command at the Privileged EXEC level or any of the CONFIG levels of the CLI:

```
BigIron# clear ipv6 neighbor ethernet 3/1
```

***Syntax:*** clear ipv6 neighbor [<ipv6-prefix>/<prefix-length> | <ipv6-address> | ethernet <port> | ve <number>]

You must specify the <ipv6-prefix> parameter in hexadecimal using 16-bit values between colons as documented in RFC 2373. You must specify the <prefix-length> parameter as a decimal value. A slash mark (/) must follow the <ipv6-prefix> parameter and precede the <prefix-length> parameter.

You must specify the <ipv6-address> parameter in hexadecimal using 16-bit values between colons as documented in RFC 2373.

The **ethernet | ve** parameter specifies the interfaces for which you can remove cache entries. If you specify an Ethernet interface, also specify the port number associated with the interface. If you specify a VE, also specify the VE number.

### Clearing IPv6 Routes from the IPv6 Route Table

You can clear all IPv6 routes or only those routes associated with a particular IPv6 prefix from the IPv6 route table and reset the routes.

For example, to clear IPv6 routes associated with the prefix 2000:7838::/32, enter the following command at the Privileged EXEC level or any of the Config levels of the CLI:

```
BigIron# clear ipv6 route 2000:7838::/32
```

***Syntax:*** clear ipv6 route [<ipv6-prefix>/<prefix-length>]

The <ipv6-prefix>/<prefix-length> parameter clears routes associated with a particular IPv6 prefix. You must specify the <ipv6-prefix> parameter in hexadecimal using 16-bit values between colons as documented in RFC 2373. You must specify the <prefix-length> parameter as a decimal value. A slash mark (/) must follow the <ipv6-prefix> parameter and precede the <prefix-length> parameter.

### Clearing IPv6 Traffic Statistics

To clear all IPv6 traffic statistics (reset all fields to zero), enter the following command at the Privileged EXEC level or any of the Config levels of the CLI:

```
BigIron(config)# clear ipv6 traffic
```

***Syntax:*** clear ipv6 traffic

### Deleting IPv6 Session Flows

To delete all flows from the IPv6 session cache, enter the following command:

```
BigIron# clear ipv6 flows
```

***Syntax:*** clear ipv6 flows

## Displaying Global IPv6 Information

You can display output for the following global IPv6 parameters:

- IPv6 cache.
- IPv6 interfaces.
- IPv6 neighbors.
- IPv6 route table.
- Local IPv6 routers.
- IPv6 TCP connections and the status of individual connections.
- IPv6 traffic statistics.
- IPv6 session flows

### Displaying IPv6 Cache Information

The IPv6 cache contains an IPv6 host table that has indices to the next hop gateway and the router interface on which the route was learned.

To display IPv6 cache information, enter the following command at any CLI level:

```
BigIron# show ipv6 cache
Total number of cache entries: 10
    IPv6 Address                        Next Hop                        Port
1   5000:2::2                           LOCAL                           tunnel 2
2   2000:4::106                         LOCAL                           ethe 3/2
3   2000:4::110                         DIRECT                          ethe 3/2
4   2002:c0a8:46a::1                    LOCAL                           ethe 3/2
5   fe80::2e0:52ff:fe99:9737            LOCAL                           ethe 3/2
6   fe80::ffff:ffff:feff:ffff           LOCAL                           loopback 2
7   fe80::c0a8:46a                      LOCAL                           tunnel 2
8   fe80::c0a8:46a                      LOCAL                           tunnel 6
9   2999::1                             LOCAL                           loopback 2
10  fe80::2e0:52ff:fe99:9700            LOCAL                           ethe 3/1
```

***Syntax:*** show ipv6 cache [<index-number> | <ipv6-prefix>/<prefix-length> | <ipv6-address> | ethernet <port> | ve <number> | tunnel <number>]

The <index-number> parameter restricts the display to the entry for the specified index number and subsequent entries.

The <ipv6-prefix>/<prefix-length> parameter restricts the display to the entries for the specified IPv6 prefix. You must specify the <ipv6-prefix> parameter in hexadecimal using 16-bit values between colons as documented in RFC 2373. You must specify the <prefix-length> parameter as a decimal value. A slash mark (/) must follow the <ipv6-prefix> parameter and precede the <prefix-length> parameter.

The **ethernet | ve | tunnel** parameter restricts the display to the entries for the specified interface. The <ipv6-address> parameter restricts the display to the entries for the specified IPv6 address. You must specify this parameter in hexadecimal using 16-bit values between colons as documented in RFC 2373.

If you specify an Ethernet interface, also specify the port number associated with the interface. If you specify a VE interface, also specify the VE number. If you specify a tunnel interface, also specify the tunnel number.

This display shows the following information:

**Table 3.1: IPv6 cache information fields**

| This Field... | Displays... |
|---|---|
| Total number of cache entries | The number of entries in the cache table. |
| IPv6 Address | The host IPv6 address. |
| Next Hop | The next hop, which can be one of the following:<br>• Direct – The next hop is directly connected to the router.<br>• Local – The next hop is originated on this router.<br>• <ipv6 address> – The IPv6 address of the next hop. |
| Port | The port on which the entry was learned. |

## Displaying IPv6 Interface Information

To display IPv6 interface information, enter the following command at any CLI level:

```
BigIron# show ipv6 interface
Routing Protocols : R - RIP  O - OSPF I - ISIS
Interface      Status     Routing  Global Unicast Address
Ethernet 3/3   down/down  R
Ethernet 3/5   down/down
Ethernet 3/17  up/up               2017::c017:101/64
Ethernet 3/19  up/up               2019::c019:101/64
VE 4           down/down
VE 14          up/up               2024::c060:101/64
Loopback 1     up/up               ::1/128
Loopback 2     up/up               2005::303:303/128
Loopback 3     up/up
```

***Syntax:*** show ipv6 interface [<interface> [<port-number> I<number>]]

The <interface> parameter displays detailed information for a specified interface. For the interface, you can specify the **Ethernet**, **loopback**, **tunnel**, or **VE** keywords. If you specify an Ethernet interface, also specify the port number associated with the interface. If you specify a loopback, tunnel, or VE interface, also specify the number associated with the interface.

This display shows the following information:

**Table 3.2: General IPv6 interface information fields**

| This Field... | Displays... |
|---|---|
| Routing protocols | A one-letter code that represents a routing protocol that can be enabled on an interface. |
| Interface | The interface type, and the port number or number of the interface. |
| Status | The status of the interface. The entry in the Status field will be either “up/up” or “down/down”. |
| Routing | The routing protocols enabled on the interface. |
| Global Unicast Address | The global unicast address of the interface. |

To display detailed information for a specific interface, enter a command such as the following at any CLI level:

```
BigIron# show ipv6 interface ethernet 3/1
Interface Ethernet 3/1 is up, line protocol is up
  IPv6 is enabled, link-local address is fe80::2e0:52ff:fe99:97
  Global unicast address(es):
  Joined group address(es):
    ff02::9
    ff02::1:ff99:9700
    ff02::2
    ff02::1
  MTU is 1500 bytes
  ICMP redirects are enabled
  ND DAD is enabled, number of DAD attempts: 3
  ND reachable time is 30 seconds
  ND advertised reachable time is 0 seconds
  ND retransmit interval is 1 seconds
  ND advertised retransmit interval is 0 seconds
  ND router advertisements are sent every 200 seconds
  ND router advertisements live for 1800 seconds
  No Inbound Access List Set
  No Outbound Access List Set
  RIP enabled
```

This display shows the following information:

**Table 3.3: Detailed IPv6 interface information fields**

| This Field... | Displays... |
|---|---|
| Interface/line protocol status | The status of interface and line protocol. If you have disabled the interface with the **disable** command, the status will be "administratively down". Otherwise, the status is either "up" or "down". |
| IPv6 status/link-local address | The status of IPv6. The status is either "enabled" or "disabled".<br>Displays the link-local address, if one is configured for the interface. |
| Global unicast address(es) | Displays the global unicast address(es), if one or more are configured for the interface. |
| Joined group address(es) | The multicast address(es) that a router interface listens for and recognizes. |
| MTU | The setting of the maximum transmission unit (MTU) configured for the IPv6 interface. The MTU is the maximum length an IPv6 packet can have to be transmitted on the interface. If an IPv6 packet is longer than an MTU, the host that originated the packet fragments the packet and transmits its contents in multiple packets that are shorter than the configured MTU. |
| ICMP | The setting of the ICMP redirect parameter for the interface. |
| ND | The setting of the various neighbor discovery parameters for the interface. |
| Access List | The inbound and outbound access lists applied to the interface. |
| Routing protocols | The routing protocols enabled on the interface. |

## Displaying IPv6 Neighbor Information

You can display the IPv6 neighbor table, which contains an entry for each IPv6 neighbor with which the router exchanges IPv6 packets.

To display the IPv6 neighbor table, enter the following command at any CLI level:

```
BigIron(config)# show ipv6 neighbor
Total number of Neighbor entries: 3

    IPv6 Address                            LinkLayer-Addr State Age Port    IsR
1   2000:4::110                             00e0.5291.bb37 REACH 20  ethe 3/1  1
2   fe80::2e0:52ff:fe91:bb37                00e0.5291.bb37 DELAY 1   ethe 3/2  1
3   fe80::2e0:52ff:fe91:bb40               00e0.5291.bb40 STALE 5930 ethe 3/3  1
```

***Syntax:*** show ipv6 neighbor [<ipv6-prefix>/<prefix-length> | <ipv6-address> | <interface> [<port> |<number>]]

The <ipv6-prefix>/<prefix-length> parameters restrict the display to the entries for the specified IPv6 prefix. You must specify the <ipv6-prefix> parameter in hexadecimal using 16-bit values between colons as documented in RFC 2373. You must specify the <prefix-length> parameter as a decimal value. A slash mark (/) must follow the <ipv6-prefix> parameter and precede the <prefix-length> parameter.

The <ipv6-address> parameter restricts the display to the entries for the specified IPv6 address. You must specify this parameter in hexadecimal using 16-bit values between colons as documented in RFC 2373.

The <interface> parameter restricts the display to the entries for the specified router interface. For this parameter, you can specify the **Ethernet** or **VE** keywords. If you specify an Ethernet interface, also specify the port number associated with the interface. If you specify a VE interface, also specify the VE number.

This display shows the following information:

**Table 3.4: IPv6 neighbor information fields**

| This Field... | Displays... |
|---|---|
| Total number of neighbor entries | The total number of entries in the IPv6 neighbor table. |
| IPv6 Address | The 128-bit IPv6 address of the neighbor. |
| Link-Layer Address | The 48-bit interface ID of the neighbor. |
| State | The current state of the neighbor. Possible states are as follows:<br>• INCOMPLETE – Address resolution of the entry is being performed.<br>• REACH – The forward path to the neighbor is functioning properly.<br>• STALE – This entry has remained unused for the maximum interval. While stale, no action takes place until a packet is sent.<br>• DELAY – This entry has remained unused for the maximum interval, and a packet was sent before another interval elapsed.<br>• PROBE – Neighbor solicitation are transmitted until a reachability confirmation is received. |
| Age | The number of seconds the entry has remained unused. If this value remains unused for the number of seconds specified by the **ipv6 nd reachable-time command** (the default is 30 seconds), the entry is removed from the table. |
| Port | The port on which the entry was learned. |
| IsR | Determines if the neighbor is a router or host:<br>0 – Indicates that the neighbor is a host.<br>1 – Indicates that the neighbor is a router. |

## Displaying the IPv6 Route Table

To display the IPv6 route table, enter the following command at any CLI level:

```
BigIron# show ipv6 route
IPv6 Routing Table - 7 entries:
Type Codes:  C - Connected, S - Static, R - RIP, O - OSPF, B - BGP, I - ISIS
Type IPv6 Prefix                Next Hop Router           Interface  Dis/Metric
C  2000:4::/64                  ::                        ethe 3/2    0/0
S  2002::/16                    ::                        tunnel 6    1/1
S  2002:1234::/32               ::                        tunnel 6    1/1
C  2002:c0a8:46a::/64           ::                        ethe 3/2    0/0
C  2999::1/128                  ::                        loopback 2  0/0
O  2999::2/128                  fe80::2e0:52ff:fe91:bb37  ethe 3/2    110/1
C  5000:2::/64                  ::                        tunnel 2    0/0
```

***Syntax:*** show ipv6 route [<ipv6-address> | <ipv6-prefix>/<prefix-length> | bgp | connect | ospf | rip | isis | static | summary]

The <ipv6-address> parameter restricts the display to the entries for the specified IPv6 address. You must specify the <ipv6-address> parameter in hexadecimal using 16-bit values between colons as documented in RFC 2373.

The <ipv6-prefix>/<prefix-length> parameters restrict the display to the entries for the specified IPv6 prefix. You must specify the <ipv6-prefix> parameter in hexadecimal using 16-bit values between colons as documented in RFC 2373. You must specify the <prefix-length> parameter as a decimal value. A slash mark (/) must follow the <ipv6-prefix> parameter and precede the <prefix-length> parameter.

The **bgp** keyword restricts the display to entries for BGP4+ routes.

The **connect** keyword restricts the display to entries for directly connected interface IPv6 routes.

The **isis** keyword restricts the display to entries for IPv6 IS-IS routes.

The **ospf** keyword restricts the display to entries for OSPFv3 routes.

The **rip** keyword restricts the display to entries for RIPng routes.

The **static** keyword restricts the display to entries for static IPv6 routes.

The **summary** keyword displays a summary of the prefixes and different route types.

The following table lists the information displayed by the **show ipv6 route** command.

**Table 3.5: IPv6 route table fields**

| This Field... | Displays... |
|---|---|
| Number of entries | The number of entries in the IPv6 route table. |
| Type | The route type, which can be one of the following:<br>• C – The destination is directly connected to the router.<br>• S – The route is a static route.<br>• R – The route is learned from RIPng.<br>• O – The route is learned from OSPFv3.<br>• B – The route is learned from BGP4+.<br>• I – The route is learned from IPv6 IS-IS. |

**Table 3.5: IPv6 route table fields (Continued)**

| This Field... | Displays... |
|---|---|
| IPv6 Prefix | The destination network of the route. |
| Next-Hop Router | The next-hop router. |
| Interface | The interface through which this router sends packets to reach the route's destination. |
| Dis/Metric | The route's administrative distance and metric value. |

To display a summary of the IPv6 route table, enter the following command at any CLI level:

```
BigIron# show ipv6 route summary
IPv6 Routing Table - 7 entries:
  4 connected, 2 static, 0 RIP, 1 OSPF, 0 BGP
  Number of prefixes:
  /16: 1 /32: 1 /64: 3 /128: 2
```

The following table lists the information displayed by the **show ipv6 route summary** command:

**Table 3.6: IPv6 route table summary fields**

| This Field... | Displays... |
|---|---|
| Number of entries | The number of entries in the IPv6 route table. |
| Number of route types | The number of entries for each route type. |
| Number of prefixes | A summary of prefixes in the IPv6 route table, sorted by prefix length. |

## Displaying Local IPv6 Routers

The Foundry device can function as an IPv6 host, instead of an IPv6 router, if you configure IPv6 addresses on its interfaces but don't enable IPv6 routing using the **ipv6 unicast-routing** command.

From the IPv6 host, you can display information about IPv6 routers to which the host is connected. The host learns about the routers through their router advertisement messages. To display information about the IPv6 routers connected to an IPv6 host, enter the following command at any CLI level:

```
BigIron# show ipv6 router
Router fe80::2e0:80ff:fe46:3431 on Ethernet 50, last update 0 min
Hops 64, Lifetime 1800 sec
Reachable time 0 msec, Retransmit time 0 msec
```

***Syntax:*** show ipv6 router

If you configure your Foundry device to function as an IPv6 router (you configure IPv6 addresses on its interfaces and enable IPv6 routing using the **ipv6 unicast-routing** command) and you enter the **show ipv6 router command**, you will receive the following output:

```
No IPv6 router in table
```

Meaningful output for this command is generated for Foundry devices configured to function as IPv6 hosts only.

This display shows the following information:

**Table 3.7: IPv6 local router information fields**

| This Field... | Displays... |
|---|---|
| Router <ipv6 address> on <interface> <port> | The IPv6 address for a particular router interface. |
| Last update | The amount of elapsed time (in minutes) between the current and previous updates received from a router. |
| Hops | The default value that should be included in the Hop Count field of the IPv6 header for outgoing IPv6 packets. The hops value applies to the router for which you are displaying information and should be followed by IPv6 hosts attached to the router. A value of 0 indicates that the router leaves this field unspecified. |
| Lifetime | The amount of time (in seconds) that the router is useful as the default router. |
| Reachable time | The amount of time (in milliseconds) that a router assumes a neighbor is reachable after receiving a reachability confirmation. The reachable time value applies to the router for which you are displaying information and should be followed by IPv6 hosts attached to the router. A value of 0 indicates that the router leaves this field unspecified. |
| Retransmit time | The amount of time (in milliseconds) between retransmissions of neighbor solicitation messages. The retransmit time value applies to the router for which you are displaying information and should be followed by IPv6 hosts attached to the router. A value of 0 indicates that the router leaves this field unspecified. |

## Displaying IPv6 TCP Information

You can display the following IPv6 TCP information:

- General information about each TCP connection on the router, including the percentage of free memory for each of the internal TCP buffers.
- Detailed information about a specified TCP connection.

To display general information about each TCP connection on the router, enter the following command at any CLI level:

```
BigIron# show ipv6 tcp connections
Local IP address:port <-> Remote IP address:port TCP state
192.168.182.110:23 <->    192.168.8.186:4933     ESTABLISHED
192.168.182.110:8218 <->  192.168.182.106:179    ESTABLISHED
192.168.182.110:8039 <->  192.168.2.119:179      SYN-SENT
192.168.182.110:8159 <->  192.168.2.102:179      SYN-SENT
2000:4::110:179 <->       2000:4::106:8222       ESTABLISHED (1440)
Total 5 TCP connections

TCP MEMORY USAGE PERCENTAGE
FREE TCB = 98 percent
FREE TCP QUEUE BUFFER = 99 percent
FREE TCP SEND BUFFER = 97 percent
FREE TCP RECEIVE BUFFER = 100 percent
FREE TCP OUT OF SEQUENCE BUFFER = 100 percent
```

***Syntax:*** show ipv6 tcp connections

This display shows the following information:

**Table 3.8: General IPv6 TCP connection fields**

| This Field... | Displays... |
|---|---|
| Local IP address:port | The IPv4 or IPv6 address and port number of the local router interface over which the TCP connection occurs. |
| Remote IP address:port | The IPv4 or IPv6 address and port number of the remote router interface over which the TCP connection occurs. |

**Table 3.8: General IPv6 TCP connection fields (Continued)**

| This Field... | Displays... |
|---|---|
| TCP state | The state of the TCP connection. Possible states include the following:<br>• LISTEN – Waiting for a connection request.<br>• SYN-SENT – Waiting for a matching connection request after having sent a connection request.<br>• SYN-RECEIVED – Waiting for a confirming connection request acknowledgment after having both received and sent a connection request.<br>• ESTABLISHED – Data can be sent and received over the connection. This is the normal operational state of the connection.<br>• FIN-WAIT-1 – Waiting for a connection termination request from the remote TCP, or an acknowledgment of the connection termination request previously sent.<br>• FIN-WAIT-2 – Waiting for a connection termination request from the remote TCP.<br>• CLOSE-WAIT – Waiting for a connection termination request from the local user.<br>• CLOSING – Waiting for a connection termination request acknowledgment from the remote TCP.<br>• LAST-ACK – Waiting for an acknowledgment of the connection termination request previously sent to the remote TCP (which includes an acknowledgment of its connection termination request).<br>• TIME-WAIT – Waiting for enough time to pass to be sure the remote TCP received the acknowledgment of its connection termination request.<br>• CLOSED – There is no connection state. |
| FREE TCB = <percentage> | The percentage of free TCP control block (TCB) space. |
| FREE TCB QUEUE BUFFER = <percentage> | The percentage of free TCB queue buffer space. |
| FREE TCB SEND BUFFER = <percentage> | The percentage of free TCB send buffer space. |
| FREE TCB RECEIVE BUFFER = <percentage> | The percentage of free TCB receive buffer space. |
| FREE TCB OUT OF SEQUENCE BUFFER = <percentage> | The percentage of free TCB out of sequence buffer space. |

To display detailed information about a specified TCP connection, enter a command such as the following at any CLI level:

```
BigIron# show ipv6 tcp status 2000:4::110 179 2000:4::106 8222
TCP: TCB = 0x217fc300
TCP: 2000:4::110:179 <-> 2000:4::106:8222: state: ESTABLISHED Port: 1
  Send: initial sequence number = 242365900
  Send: first unacknowledged sequence number = 242434080
  Send: current send pointer = 242434080
  Send: next sequence number to send = 242434080
  Send: remote received window = 16384
  Send: total unacknowledged sequence number = 0
  Send: total used buffers 0
  Receive: initial incoming sequence number = 740437769
  Receive: expected incoming sequence number = 740507227
  Receive: received window = 16384
  Receive: bytes in receive queue = 0
  Receive: congestion window = 1459
```

***Syntax:*** show ipv6 tcp status <local-ip-address> <local-port-number> <remote-ip-address> <remote-port-number>

The <local-ip-address> parameter can be the IPv4 or IPv6 address of the local interface over which the TCP connection is taking place.

The <local-port-number> parameter is the local port number over which a TCP connection is taking place.

The <remote-ip-address> parameter can be the IPv4 or IPv6 address of the remote interface over which the TCP connection is taking place.

The <remote-port-number> parameter is the local port number over which a TCP connection is taking place.

This display shows the following information:

**Table 3.9: Specific IPv6 TCP connection fields**

| This Field... | Displays... |
|---|---|
| TCB = <location> | The location of the TCB. |
| <local-ip-address> <local-port-number> <remote-ip-address> <remote-port-number> <state> <port> | This field provides a general summary of the following:<br>• The local IPv4 or IPv6 address and port number.<br>• The remote IPv4 or IPv6 address and port number.<br>• The state of the TCP connection. For information on possible states, see Table  on page 3-30.<br>• The port numbers of the local interface. |
| Send: initial sequence number = <number> | The initial sequence number sent by the local router. |
| Send: first unacknowledged sequence number = <number> | The first unacknowledged sequence number sent by the local router. |

**Table 3.9: Specific IPv6 TCP connection fields (Continued)**

| This Field... | Displays... |
|---|---|
| Send: current send pointer = <number> | The current send pointer. |
| Send: next sequence number to send = <number> | The next sequence number sent by the local router. |
| Send: remote received window = <number> | The size of the remote received window. |
| Send: total unacknowledged sequence number = <number> | The total number of unacknowledged sequence numbers sent by the local router. |
| Send: total used buffers <number> | The total number of buffers used by the local router in setting up the TCP connection. |
| Receive: initial incoming sequence number = <number> | The initial incoming sequence number received by the local router. |
| Receive: expected incoming sequence number = <number> | The incoming sequence number expected by the local router. |
| Receive: received window = <number> | The size of the local router's receive window. |
| Receive: bytes in receive queue = <number> | The number of bytes in the local router's receive queue. |
| Receive: congestion window = <number> | The size of the local router's receive congestion window. |

## Displaying IPv6 Traffic Statistics

To display IPv6 traffic statistics, enter the following command at any CLI level:

```
BigIron# show ipv6 traffic
IP6 Statistics

  36947 received, 66818 sent, 0 forwarded, 36867 delivered, 0 rawout
  0 bad vers, 23 bad scope, 0 bad options, 0 too many hdr
  0 no route, 0 can't forward, 0 redirect sent
  0 frag recv, 0 frag dropped, 0 frag timeout, 0 frag overflow
  0 reassembled, 0 fragmented, 0 ofragments, 0 can't frag
  0 too short, 0 too small, 11 not member
  0 no buffer, 66819 allocated, 21769 freed
  0 forward cache hit, 46 forward cache miss

ICMP6 Statistics
Received:
  0 dest unreach, 0 pkt too big, 0 time exceeded, 0 param prob
  2 echo req, 1 echo reply, 0 mem query, 0 mem report, 0 mem red
  0 router soli, 2393 router adv, 106 nei soli, 3700 nei adv, 0 redirect
  0 bad code, 0 too short, 0 bad checksum, 0 bad len
  0 reflect, 0 nd toomany opt, 0 badhopcount
Sent:
  0 dest unreach, 0 pkt too big, 0 time exceeded, 0 param prob
  1 echo req, 2 echo reply, 0 mem query, 0 mem report, 0 mem red
  0 router soli, 2423 router adv, 3754 nei soli, 102 nei adv, 0 redirect
  0 error, 0 can't send error, 0 too freq
Sent Errors:
  0 unreach no route, 0 admin, 0 beyond scope, 0 address, 0 no port
  0 pkt too big, 0 time exceed transit, 0 time exceed reassembly
  0 param problem header, 0 nextheader, 0 option, 0 redirect, 0 unknown

UDP Statistics
  470 received, 7851 sent, 6 no port, 0 input errors

TCP Statistics
  57913 active opens, 0 passive opens, 57882 failed attempts
  159 active resets, 0 passive resets, 0 input errors
  565189 in segments, 618152 out segments, 171337 retransmission
```

***Syntax:*** show ipv6 traffic

This display shows the following information:

**Table 3.10: IPv6 traffic statistics fields**

| This Field... | Displays... |
|---|---|
| **IPv6 statistics** | |
| received | The total number of IPv6 packets received by the router. |
| sent | The total number of IPv6 packets originated and sent by the router. |
| forwarded | The total number of IPv6 packets received by the router and forwarded to other routers. |

**Table 3.10: IPv6 traffic statistics fields (Continued)**

| This Field... | Displays... |
|---|---|
| delivered | The total number of IPv6 packets delivered to the upper layer protocol. |
| rawout | This information is used by Foundry Technical Support. |
| bad vers | The number of IPv6 packets dropped by the router because the version number is not 6. |
| bad scope | The number of IPv6 packets dropped by the router because of a bad address scope. |
| bad options | The number of IPv6 packets dropped by the router because of bad options. |
| too many hdr | The number of IPv6 packets dropped by the router because the packets had too many headers. |
| no route | The number of IPv6 packets dropped by the router because there was no route. |
| can't forward | The number of IPv6 packets the router could not forward to another router. |
| redirect sent | This information is used by Foundry Technical Support. |
| frag recv | The number of fragments received by the router. |
| frag dropped | The number of fragments dropped by the router. |
| frag timeout | The number of fragment timeouts that occurred. |
| frag overflow | The number of fragment overflows that occurred. |
| reassembled | The number of fragmented IPv6 packets that the router reassembled. |
| fragmented | The number of IPv6 packets fragmented by the router to accommodate the MTU of this router or of another device. |
| ofragments | The number of output fragments generated by the router. |
| can't frag | The number of IPv6 packets the router could not fragment. |
| too short | The number of IPv6 packets dropped because they are too short. |
| too small | The number of IPv6 packets dropped because they don't have enough data. |
| not member | The number of IPv6 packets dropped because the recipient is not a member of a multicast group. |
| no buffer | The number of IPv6 packets dropped because there is no buffer available. |
| forward cache miss | The number of IPv6 packets received for which there is no corresponding cache entry. |
| **ICMP6 statistics**<br>Some ICMP statistics apply to both Received and Sent, some apply to Received only, some apply to Sent only, and some apply to Sent Errors only. | |
| **Applies to Received and Sent** | |
| dest unreach | The number of Destination Unreachable messages sent or received by the router. |
| pkt too big | The number of Packet Too Big messages sent or received by the router. |

**Table 3.10: IPv6 traffic statistics fields (Continued)**

| This Field... | Displays... |
|---|---|
| time exceeded | The number of Time Exceeded messages sent or received by the router. |
| param prob | The number of Parameter Problem messages sent or received by the router. |
| echo req | The number of Echo Request messages sent or received by the router. |
| echo reply | The number of Echo Reply messages sent or received by the router. |
| mem query | The number of Group Membership Query messages sent or received by the router. |
| mem report | The number of Membership Report messages sent or received by the router. |
| mem red | The number of Membership Reduction messages sent or received by the router. |
| router soli | The number of Router Solicitation messages sent or received by the router. |
| router adv | The number of Router Advertisement messages sent or received by the router. |
| nei soli | The number of Neighbor Solicitation messages sent or received by the router. |
| nei adv | The number of Router Advertisement messages sent or received by the router. |
| redirect | The number of redirect messages sent or received by the router. |
| **Applies to Received Only** | |
| bad code | The number of Bad Code messages received by the router. |
| too short | The number of Too Short messages received by the router. |
| bad checksum | The number of Bad Checksum messages received by the router. |
| bad len | The number of Bad Length messages received by the router. |
| nd toomany opt | The number of Neighbor Discovery Too Many Options messages received by the router. |
| badhopcount | The number of Bad Hop Count messages received by the router. |
| **Applies to Sent Only** | |
| error | The number of Error messages sent by the router. |
| can’t send error | The number of times the node encountered errors in ICMP error messages. |
| too freq | The number of times the node has exceeded the frequency of sending error messages. |
| **Applies to Sent Errors Only** | |
| unreach no route | The number of Unreachable No Route errors sent by the router. |
| admin | The number of Admin errors sent by the router. |
| beyond scope | The number of Beyond Scope errors sent by the router. |
| address | The number of Address errors sent by the router. |
| no port | The number of No Port errors sent by the router. |

**Table 3.10: IPv6 traffic statistics fields (Continued)**

| This Field... | Displays... |
|---|---|
| pkt too big | The number of Packet Too Big errors sent by the router. |
| time exceed transit | The number of Time Exceed Transit errors sent by the router. |
| time exceed reassembly | The number of Time Exceed Reassembly errors sent by the router. |
| param problem header | The number of Parameter Problem Header errors sent by the router. |
| nextheader | The number of Next Header errors sent by the router. |
| option | The number of Option errors sent by the router. |
| redirect | The number of Redirect errors sent by the router. |
| unknown | The number of Unknown errors sent by the router. |
| **UDP statistics** | |
| received | The number of UDP packets received by the router. |
| sent | The number of UDP packets sent by the router. |
| no port | The number of UDP packets dropped because the packet did not contain a valid UDP port number. |
| input errors | This information is used by Foundry Technical Support. |
| **TCP statistics** | |
| active opens | The number of TCP connections opened by the router by sending a TCP SYN to another device. |
| passive opens | The number of TCP connections opened by the router in response to connection requests (TCP SYNs) received from other devices. |
| failed attempts | This information is used by Foundry Technical Support. |
| active resets | The number of TCP connections the router reset by sending a TCP RESET message to the device at the other end of the connection. |
| passive resets | The number of TCP connections the router reset because the device at the other end of the connection sent a TCP RESET message. |
| input errors | This information is used by Foundry Technical Support. |
| in segments | The number of TCP segments received by the router. |
| out segments | The number of TCP segments sent by the router. |
| retransmission | The number of segments that the router retransmitted because the retransmission timer for the segment had expired before the device at the other end of the connection had acknowledged receipt of the segment. |

## Displaying IPv6 Session Flows

If you want to display the contents of an IPv6 session cache, enter the following command:

```
BigIron# show ipv6 flows
```

***Syntax:*** show ipv6 flows [<source-ipv6-prefix/prefix-length> | any | host <source-ipv6_address> <destination-ipv6-prefix/prefix-length> | any | host <destination-ipv6-address>]

If you do not specify a source or destination, all IPv6 flows are displayed.

Enter a value for <source-ipv6-prefix>/<prefix-length> or <destination-ipv6-prefix>/<prefix-length> to specify a source or destination prefix and prefix length that a flow must match to be included in the display.

Enter **any** for source or destination if a flow can have any source or any destination to be included in the display.

The **host** <source-ipv6-address> and **host** <destination-ipv6-address> parameters allow you specify a source or destination host IPv6 address that a flow must match to be included in the display.

**EXAMPLES:**

To show all IPv6 flows, enter the following command:

```
BigIron# show ipv6 flows
```

To show all IPv6 flows with any IPv6 source and any IPv6 destination addresses, enter the following command:

```
BigIron# show ipv6 flows any any
```

To show all IPv6 flows that match the source prefix 4000::/16 and any destination address, enter the following command:

```
BigIron# show ipv6 flows 4000::/16 any
```

To show all IPv6 flows that have any source address but only a destination address of host 5020::30, enter the following command:

```
BigIron# show ipv6 flows any host 5020::30
```

To show all IPv6 flows that have the source address of host 4050::30 and the destination address of host 5020::30, enter the following command:

```
BigIron# show ipv6 flows host 4050::30 host 5020::30
```

The following is an example of what is displayed when you enter the **show ipv6 flows** command:

```
BigIron# show ipv6 flows

ipv6 flows count: 6
A:Ack D:Deny E:Estab F:Fin P:Psh Pe:Permit R:Rst U:urg Fr:Fragment
Sr:SRouted
SourceAddress                          DestinationAddress
  Protocol SrcPort/IcmpType DestPort/IcmpCode Dscp FlowLabel  Flags     Age
3001::3                                3020::160
 icmp      128             0                 0    0          Pe        4
3001::3                                3020::160
 tcp       telnet          3456              0    0          DAR       3
3001::3                                3020::160
 tcp       telnet          3456              0    0          DAS       3
3001::3                                3020::160
 icmp      129             0                 0    0          Pe        8
3001::3                                3020::160
 tcp       3456            telnet            0    0          DAR       9
3001::3                                3020::165
 icmp      128             0                 0    0          Pe        4
```

The first line (ipv6 flows count) shows the number of flows included on the display.

The next line defines the flags used in the display.

Information for each flow on the display appears on two lines in the following sequence:

- Source Address – Source address of the flow.
- Destination Address – Destination address of the flow.

- Protocol – Protocol in the flow.
- SrcPort/IcmpType – Either the source TCP/UDP port or the ICMP type of the flow.
- DestPort/IcmpCode – Either the destination TCP/UDP port or the ICMP code of the flow.
- Dscp – DSCP value in the flow.
- FlowLabel – Value in the flow label field of the IPv6 packet header.
- Flags – Status of the flow, which can be a combination of different flag types. For example, DAR means the flow was denied (D), acknowledged (A), and reset (R).
- Age – Age of the flow.

**NOTE:** The life of an idle flow is 50 seconds.

# Chapter 4
# Configuring Static IPv6 Routes

This chapter describes how to configure a static IPv6 route. A ***static IPv6 route*** is a manually configured route that creates a path between two IPv6 routers. A static IPv6 route is similar to a static IPv4 route. Static IPv6 routes have advantages and disadvantages; for example, a static IPv6 route does not generate updates, which reduces processing time for an IPv6 router. Conversely, if a static IPv6 route fails or if you want to change your network topology, you might need to manually reconfigure the static IPv6 route.

## Configuring a Static IPv6 Route

You can configure a static IPv6 route to be redistributed into a routing protocol, but you cannot redistribute routes learned by a routing protocol into the static IPv6 routing table.

Before configuring a static IPv6 route, you must enable the forwarding of IPv6 traffic on the router using the **ipv6 unicast-routing** command and enable IPv6 on at least one interface by configuring an IPv6 address or explicitly enabling IPv6 on that interface. For more information on performing these configuration tasks, see “Configuring Basic IPv6 Connectivity” on page 3-1.

To configure a static IPv6 route for a destination network with the prefix 8eff::0/32, a next-hop gateway with the global address 4fee:2343:0:ee44::1, and an administrative distance of 110, enter the following command:

```
BigIron(config)# ipv6 route 8eff::0/32 4fee:2343:0:ee44::1 distance 110
```

***Syntax:*** ipv6 route <dest-ipv6-prefix>/<prefix-length> <next-hop-ipv6-address> [<metric>] [distance <number>]

To configure a static IPv6 route for a destination network with the prefix 8eff::0/32 and a next-hop gateway with the link-local address fe80::1, that the router can access through Ethernet interface 3/1, enter the following command:

```
BigIron(config)# ipv6 route 8eff::0/32 ethernet 1 fe80::1
```

***Syntax:*** ipv6 route <dest-ipv6-prefix>/<prefix-length> <interface> <port> <next-hop-ipv6-address> [<metric>] [distance <number>]

To configure a static IPv6 route for a destination network with the prefix 8eff::0/32 and a next-hop gateway that the router can access through tunnel 1, enter the following command:

```
BigIron(config)# ipv6 route 8eff::0/32 tunnel 1
```

***Syntax:*** ipv6 route <dest-ipv6-prefix>/<prefix-length> <interface> <port> [<metric>] [distance <number>]

Table 4.1 describes the parameters associated with this command and indicates the status of each parameter.

**Table 4.1: Static IPv6 route parameters**

| Parameter | Configuration Details | Status |
|---|---|---|
| The IPv6 prefix and prefix length of the route's destination network. | You must specify the <dest-ipv6-prefix> parameter in hexadecimal using 16-bit values between colons as documented in RFC 2373.<br>You must specify the <prefix-length> parameter as a decimal value. A slash mark (/) must follow the <ipv6-prefix> parameter and precede the <prefix-length> parameter. | Mandatory for all static IPv6 routes. |
| The route's next-hop gateway, which can be one of the following:<br>• The IPv6 address of a next-hop gateway.<br>• A tunnel interface. | You can specify the next-hop gateway as one of the following types of IPv6 addresses:<br>• A global address<br>• A link-local address<br>If you specify a global address, you do not need to specify any additional parameters for the next-hop gateway.<br>If you specify a link-local address, you must also specify the interface through which to access the address. You can specify one of the following interfaces:<br>• An Ethernet interface<br>• A tunnel interface<br>• A virtual interface (VE)<br>If you specify an Ethernet interface, also specify the port number associated with the interface. If you specify a VE or tunnel interface, also specify the VE or tunnel number.<br>You can also specify the next-hop gateway as a tunnel interface. If you specify a tunnel interface, also specify the tunnel number. | Mandatory for all static IPv6 routes. |
| The route's metric. | You can specify a value from 1 – 16. | Optional for all static IPv6 routes. (The default metric is 1.) |
| The route's administrative distance. | You must specify the **distance** keyword and any numerical value. | Optional for all static IPv6 routes. (The default administrative distance is 1.) |

A metric is a value that the router uses when comparing this route to other static routes in the IPv6 static route table that have the same destination. The metric applies only to routes that the router has already placed in the IPv6 static route table.

The administrative distance is a value that the router uses to compare this route with routes from other route sources that have the same destination. (The router performs this comparison before placing a route in the IPv6

route table.) This parameter does not apply to routes that are already in the IPv6 route table. In general, a low administrative distance indicates a preferred route. By default, static routes take precedence over routes learned by routing protocols. If you want a dynamic route to be chosen over a static route, you can configure the static route with a higher administrative distance than the dynamic route.

# Chapter 5
# Configuring RIPng

Routing Information Protocol (RIP) is an IP route exchange protocol that uses a distance vector (a number representing a distance) to measure the cost of a given route. RIP uses a hop count as its cost or metric.

IPv6 RIP, known as ***Routing Information Protocol Next Generation*** or ***RIPng***, functions similarly to IPv4 RIP version 2. RIPng supports IPv6 addresses and prefixes.

In addition, Foundry implements some new commands that are specific to RIPng. This chapter describes the commands that are specific to RIPng. This section does not describe commands that apply to both IPv4 RIP and RIPng. For more information about these commands, see the *Foundry Router Configuration Guide*.

RIPng maintains a ***Routing Information Database (RIB)***, which is a local route table. The local RIB contains the lowest-cost IPv6 routes learned from other RIP routers. In turn, RIPng attempts to add routes from its local RIB into the main IPv6 route table.

This chapter describes the following:

- How to configure RIPng
- How to clear RIPng information from the RIPng route table
- How to display RIPng information and statistics

## Configuring RIPng

To configure RIPng, you must do the following:

- Enable RIPng globally on the Foundry device and on individual router interfaces

The following configuration tasks are optional:

- Change the default settings of RIPng timers
- Configure how the Foundry device learns and advertises routes
- Configure which routes are redistributed into RIPng from other sources
- Configure how the Foundry device distributes routes via RIPng
- Configure poison reverse parameters

### Enabling RIPng

Before configuring the Foundry device to run RIPng, you must do the following:

- Enable the forwarding of IPv6 traffic on the Foundry device using the **ipv6 unicast-routing** command.

- Enable IPv6 on each interface over which you plan to enable RIPng. You enable IPv6 on an interface by configuring an IPv6 address or explicitly enabling IPv6 on that interface.

For more information about performing these configuration tasks, see “Configuring Basic IPv6 Connectivity” on page 3-1.

By default, RIPng is disabled. To enable RIPng, you must enable it globally on the Foundry device and also on individual router interfaces.

**NOTE:** Enabling RIPng globally on the Foundry device does not enable it on individual router interfaces.

To enable RIPng globally, enter the following command:

```
BigIron(config-rip-router)#ipv6 router rip
BigIron(config-ripng-router)#
```

After you enter this command, the Foundry device enters the RIPng configuration level, where you can access several commands that allow you to configure RIPng.

***Syntax:*** [no] ipv6 router rip

To disable RIPng globally, use the **no** form of this command.

After enabling RIPng globally, you must enable it on individual router interfaces. You can enable it on physical as well as virtual routing interfaces. For example, to enable RIPng on Ethernet interface 3/1, enter the following commands:

```
BigIron(config)# interface ethernet 3/1
BigIron(config-if-e100-3/1)# ipv6 rip enable
```

***Syntax:*** [no] ipv6 rip enable

To disable RIPng on an individual router interface, use the **no** form of this command.

## Configuring RIPng Timers

Table 5.1 describes the RIPng timers and provides their defaults.

**Table 5.1: RIPng timers**

| Timer | Description | Default |
|---|---|---|
| Update | Amount of time (in seconds) between RIPng routing updates. | 30 seconds. |
| Timeout | Amount of time (in seconds) after which a route is considered unreachable. | 180 seconds. |
| Hold-down | Amount of time (in seconds) during which information about other paths is ignored. | 180 seconds. |
| Garbage-collection | Amount of time (in seconds) after which a route is removed from the routing table. | 120 seconds. |

You can adjust these timers for RIPng. Before doing so, keep the following caveats in mind:

- If you adjust these RIPng timers, Foundry strongly recommends setting the same timer values for all routers and access servers in the network.
- Setting a shorter update interval can cause the routers to spend excessive updating to the IPv6 route table.
- Foundry recommends setting the timeout timer value to at least three times the value of the update timer.
- Foundry recommends a shorter hold-down timer interval, because a longer interval can cause delays in RIPng convergence.

The following example sets updates to be broadcast every 45 seconds. If a route is not heard from in 135 seconds, the route is declared unusable. Further information is suppressed for an additional 10 seconds. Assuming no updates, the route is flushed from the routing table 20 seconds after the end of the hold-down period.

```
BigIron(config)# ipv6 router rip
BigIron(config-ripng-router)# timers 45 135 10 20
```

***Syntax:*** [no] timers <update-timer> <timeout-timer> <hold-down-timer> <garbage-collection-timer>

Possible values for the timers are as follows:

- Update timer: 3 – 65535 seconds
- Timeout timer: 9 – 65535 seconds
- Hold-down timer: 9 – 65535 seconds
- Garbage-collection timer: 9 – 65535 seconds

**NOTE:** You must enter a value for each timer, even if you want to retain the current setting of a particular timer.

To return to the default values of the RIPng timers, use the **no** form of this command.

## Configuring Route Learning and Advertising Parameters

You can configure the following learning and advertising parameters:

- Learning and advertising of RIPng default routes
- Advertising of IPv6 address summaries
- Metric of routes learned and advertised on a router interface

### Configuring Default Route Learning and Advertising

By default, the Foundry device does not learn IPv6 default routes (::/0). You can originate default routes into RIPng, which causes individual router interfaces to include the default routes in their updates. When configuring the origination of the default routes, you can also do the following:

- Suppress all other routes from the updates
- Include all other routes in the updates

For example, to originate default routes in RIPng and suppress all other routes in updates sent from Ethernet interface 3/1, enter the following commands:

```
BigIron(config)# interface ethernet 3/1
BigIron(config-if-e100-3/1)# ipv6 rip default-information only
```

To originate IPv6 default routes and include all other routes in updates sent from Ethernet interface 3/1, enter the following commands:

```
BigIron(config)# interface ethernet 3/1
BigIron(config-if-e100-3/1)# ipv6 rip default-information originate
```

***Syntax:*** [no] ipv6 rip default-information only | originate

The **only** keyword originates the default routes and suppresses all other routes from the updates.

The **originate** keyword originates the default routes and includes all other routes in the updates.

To remove the explicit default routes from RIPng and suppress advertisement of these routes, use the **no** form of this command.

### Advertising IPv6 Address Summaries

You can configure RIPng to advertise a summary of IPv6 addresses from a router interface and to specify an IPv6 prefix that summarizes the routes.

If a route's prefix length matches the value specified in the **ipv6 rip summary-address** command, RIPng advertises the prefix specified in the **ipv6 rip summary-address** command instead of the original route.

For example, to advertise the summarized prefix 2001:469e::/36 instead of the IPv6 address 2001:469e:0:adff:8935:e838:78:e0ff with a prefix length of 64 bits from Ethernet interface 3/1, enter the following commands:

```
BigIron(config)# interface ethernet 3/1
BigIron(config-if-e100-3/1)# ipv6 address 2001:469e:0:adff:8935:e838:78:
e0ff /64
BigIron(config-if-e100-3/1)# ipv6 rip summary-address 2001:469e::/36
```

***Syntax:*** [no] ipv6 rip summary-address <ipv6-prefix>/<prefix-length>

You must specify the <ipv6-prefix> parameter in hexadecimal using 16-bit values between colons as documented in RFC 2373.

You must specify the <prefix-length> parameter as a decimal value. A slash mark (/) must follow the <ipv6-prefix> parameter and precede the <prefix-length> parameter.

To stop the advertising of the summarized IPv6 prefix, use the **no** form of this command.

### Changing the Metric of Routes Learned and Advertised on an Interface

A router interface increases the metric of an incoming RIPng route it learns by an offset (the default is one). The Foundry device then places the route in the route table. When the Foundry device sends an update, it advertises the route with the metric plus the default offset of zero in an outgoing update message.

You can change the metric offset an individual interface adds to a route learned by the interface or advertised by the interface. For example, to change the metric offset for incoming routes learned by Ethernet interface 3/1 to one and the metric offset for outgoing routes advertised by the interface to three, enter the following commands:

```
BigIron(config)# interface ethernet 3/1
BigIron(config-if-e100-3/1)# ipv6 rip metric-offset 1
BigIron(config-if-e100-3/1)# ipv6 rip metric-offset out 3
```

In this example, if Ethernet interface 3/1 learns about an incoming route, it will increase the incoming metric by two (the default offset of 1 and the additional offset of 1 as specified in this example). If Ethernet interface 3/1 advertises an outgoing route, it will increase the metric by 3 as specified in this example.

***Syntax:*** [no] ipv6 rip metric-offset [out] <1 – 16>

To return the metric offset to its default value, use the **no** form of this command.

## Redistributing Routes Into RIPng

You can configure the Foundry device to redistribute routes from the following sources into RIPng:

- IPv6 static routes
- Directly connected IPv6 networks
- BGP4+
- IPv6 IS-IS
- OSPFv3

When you redistribute a route from BGP4+, IPv6 IS-IS, or OSPFv3 into RIPng, the Foundry device can use RIPng to advertise the route to its RIPng neighbors.

When configuring the Foundry device to redistribute routes, such as BGP4+ routes, you can optionally specify a metric for the redistributed routes. If you do not explicitly configure a metric, the default metric value of one is used.

For example, to redistribute OSPFv3 routes into RIPng, enter the following command:

```
BigIron(config)# ipv6 router rip
BigIron(config-ripng-router)# redistribute ospf
```

***Syntax:*** redistribute bgp | connected | isis | ospf | static [metric <number>]

For the metric, specify a numerical value that is consistent with RIPng.

## Controlling Distribution of Routes Via RIPng

You can create a prefix list and then apply it to RIPng routing updates that are received or sent on a router interface. Performing this task allows you to control the distribution of routes via RIPng.

For example, to permit the inclusion of routes with the prefix 2001::/16 in RIPng routing updates sent from Ethernet interface 3/1, enter the following commands:

```
BigIron(config)# ipv6 prefix-list routesfor2001 permit 2001::/16
BigIron(config)# ipv6 router rip
BigIron(config-ripng-router)# distribute-list prefix-list routesfor2001 out
ethernet 3/1
```

To deny prefix lengths greater than 64 bits in routes that have the prefix 3EE0:A99::/64 and allow all other routes received on tunnel interface 3/1, enter the following commands:

```
BigIron(config)# ipv6 prefix-list 3ee0routes deny 3ee0:a99::/64 le 128
BigIron(config)# ipv6 prefix-list 3ee0routes permit ::/0 ge 0 le 128
BigIron(config)# ipv6 router rip
BigIron(config-ripng-router)# distribute-list prefix-list 3ee0routes in
tunnel 1
```

For information about prefix lists, including the syntax of the **ipv6 prefix-list** command, see “Configuring an IPv6 Prefix List” on page 11-1.

***Syntax:*** [no] distribute-list prefix-list <name> in | out <interface> <port>

The <name> parameter indicates the name of the prefix list generated using the **ipv6 prefix-list** command.

The **in** keyword indicates that the prefix list is applied to incoming routing updates on the specified interface.

The **out** keyword indicates that the prefix list is applied to outgoing routing updates on the specified interface.

For the <interface> parameter, you can specify the **ethernet**, **loopback**, **ve**, or **tunnel** keywords. If you specify an Ethernet interface, also specify the port number associated with the interface. If you specify a VE or tunnel interface, also specify the VE or tunnel number.

To remove the distribution list, use the **no** form of this command.

## Configuring Poison Reverse Parameters

By default, poison reverse is disabled on a RIPng router. If poison reverse is enabled, RIPng advertises routes it learns from a particular interface over that same interface with a metric of 16, which means that the route is unreachable.

If poison reverse is enabled on the RIPng router, it takes precedence over split horizon (if it is also enabled).

To enable poison reverse on the RIPng router, enter the following commands:

```
BigIron(config)# ipv6 router rip
BigIron(config-ripng-router)# poison-reverse
```

***Syntax:*** [no] poison-reverse

To disable poison-reverse, use the **no** version of this command.

By default, if a RIPng interface goes down, the Foundry device does not send a triggered update for the interface’s IPv6 networks.

To better handle this situation, you can configure a RIPng router to send a triggered update containing the local routes of the disabled interface with an unreachable metric of 16 to the other RIPng routers in the routing domain. You can enable the sending of a triggered update by entering the following commands:

```
BigIron(config)# ipv6 router rip
```

```
BigIron(config-ripng-router)# poison-local-routes
```

***Syntax:*** [no] poison-local-routes

To disable the sending of a triggered update, use the **no** version of this command.

# Clearing RIPng Routes from IPv6 Route Table

To clear all RIPng routes from the RIPng route table and the IPv6 main route table and reset the routes, enter the following command at the Privileged EXEC level or any of the Config levels of the CLI:

```
BigIron# clear ipv6 rip routes
```

***Syntax:*** clear ipv6 rip routes

# Displaying RIPng Information

You can display the following RIPng information:

- RIPng configuration
- RIPng routing table

## Displaying RIPng Configuration

To display RIPng configuration information, enter the following command at any CLI level:

```
BigIron# show ipv6 rip
 IPv6 rip enabled, port 521
     Administrative distance is 120
     Updates every 30 seconds, expire after 180
     Holddown lasts 180 seconds, garbage collect after 120
     Split horizon is on; poison reverse is off
     Default routes are not generated
     Periodic updates 0, trigger updates 0
     Distribute List, Inbound : Not set
     Distribute List, Outbound : Not set
     Redistribute: CONNECTED
```

***Syntax:*** show ipv6 rip

This display shows the following information:

**Table 5.2: RIPng configuration fields**

| This Field... | Displays... |
|---|---|
| IPv6 RIP status/port | The status of RIPng on the Foundry device. Possible status is “enabled” or “disabled.”<br>The UDP port number over which RIPng is enabled. |
| Administrative distance | The setting of the administrative distance for RIPng. |
| Updates/expiration | The settings of the RIPng update and timeout timers. |
| Holddown/garbage collection | The settings of the RIPng hold-down and garbage-collection timers. |
| Split horizon/poison reverse | The status of the RIPng split horizon and poison reverse features. Possible status is “on” or “off.” |

**Table 5.2: RIPng configuration fields**

| This Field... | Displays... |
|---|---|
| Default routes | The status of RIPng default routes. |
| Periodic updates/trigger updates | The number of periodic updates and triggered updates sent by the RIPng router. |
| Distribution lists | The inbound and outbound distribution lists applied to RIPng. |
| Redistribution | The types of IPv6 routes redistributed into RIPng. The types can include the following:<br>• STATIC – IPv6 static routes are redistributed into RIPng.<br>• CONNECTED – Directly connected IPv6 networks are redistributed into RIPng.<br>• BGP – BGP4+ routes are redistributed into RIPng.<br>• ISIS – IPv6 IS-IS routes are redistributed into RIPng.<br>• OSPF – OSPFv3 routes are redistributed into RIPng. |

## Displaying RIPng Routing Table

To display the RIPng routing table, enter the following command at any CLI level:

```
BigIron# show ipv6 rip route
IPv6 RIP Routing Table - 4 entries:
 2000:4::/64, from ::, null   (0)
          CONNECTED, metric 1, tag 0, timers: none
 2002:c0a8:46a::/64, from ::, null   (1)
          CONNECTED, metric 1, tag 0, timers: none
 2999::1/128, from ::, null   (2)
          CONNECTED, metric 1, tag 0, timers: none
 5000:2::/64, from ::, null   (3)
          CONNECTED, metric 1, tag 0, timers: none
```

***Syntax:*** show ipv6 rip route [<ipv6-prefix>/<prefix-length> | <ipv6-address>]

The <ipv6-prefix>/<prefix-length> parameters restrict the display to the entries for the specified IPv6 prefix. You must specify the <ipv6-prefix> parameter in hexadecimal using 16-bit values between colons as documented in RFC 2373. You must specify the <prefix-length> parameter as a decimal value. A slash mark (/) must follow the <ipv6-prefix> parameter and precede the <prefix-length> parameter.

The <ipv6-address> parameter restricts the display to the entries for the specified IPv6 address. You must specify this parameter in hexadecimal using 16-bit values between colons as documented in RFC 2373.

This display shows the following information:

**Table 5.3: RIPng routing table fields**

| This Field... | Displays... |
|---|---|
| RIPng Routing Table entries | The total number of entries in the RIPng routing table. |
| <ipv6-prefix>/<prefix-length> | The IPv6 prefix and prefix length. |
| <ipv6-address> | The IPv6 address. |

**Table 5.3: RIPng routing table fields**

| This Field... | Displays... |
|---|---|
| Next-hop router | The next-hop router for this Foundry device. If :: appears, the route is originated locally. |
| Interface | The interface name. If “null” appears, the interface is originated locally. |
| Source of route | The source of the route information. The source can be one of the following:<br>• RIP – routes learned by RIPng.<br>• CONNECTED – IPv6 routes redistributed from directly connected networks.<br>• STATIC – IPv6 static routes are redistributed into RIPng.<br>• BGP – BGP4+ routes are redistributed into RIPng.<br>• ISIS – IPv6 IS-IS routes are redistributed into RIPng.<br>• OSPF – OSPFv3 routes are redistributed into RIPng. |
| Metric <number> | The cost of the route. The <number> parameter indicates the number of hops to the destination. |
| Tag <number> | The tag value of the route. |
| Timers: | Indicates if the hold-down timer or the garbage-collection timer is set. |

# Chapter 6
# Configuring OSPF Version 3

Open Shortest Path First (OSPF) is a link-state routing protocol. OSPF uses link-state advertisements (LSAs) to update neighboring routers about its interfaces and information on those interfaces. The router floods LSAs to all neighboring routers to update them about the interfaces. Each router maintains an identical database that describes its area topology to help a router determine the shortest path between it and any neighboring router.

This chapter describes the following:

- The differences between OSPF versions 2 and 3
- The link state advertisement types for OSPF version 3
- How to configure OSPF version 3
- How to display OSPF version 3 information and statistics

## OSPF Version 3

IPv6 supports OSPF version 3 (OSPFv3), which functions similarly to OSPF version 2, the current version that IPv4 supports, except for the following enhancements:

- Support for IPv6 addresses and prefixes.
- In general, you can configure several IPv6 addresses on a router interface. OSPFv3 imports all or none of the address prefixes configured on a router interface. You cannot select which addresses to import.
- You can run one instance of OSPF version 2 and one instance of OSPFv3 concurrently on a link.
- IPv6 link state advertisements (LSAs).

In addition, Foundry implements some new commands that are specific to OSPFv3. This section describes the commands that are specific to OSPFv3.

**NOTE:** Although OSPF versions 2 and 3 function similarly to each other, Foundry has implemented the user interface for each version independently of each other. Therefore, any configuration of OSPF version 2 features will not affect the configuration of OSPFv3 features and vice versa.

## Link State Advertisement Types for OSPFv3

OSPFv3 supports the following types of LSAs:

- Router LSAs (Type 1)

- Network LSAs (Type 2)
- Interarea-prefix LSAs for ABRs (Type 3)
- Interarea-router LSAs for ASBRs (Type 4)
- Autonomous system external LSAs (Type 5)
- Link LSAs (Type 8)
- Intra-area prefix LSAs (Type 9)

For more information about these LSAs, see RFC 2740.

# Configuring OSPFv3

To configure OSPFv3, you must do the following:

- Enable OSPFv3 globally.
- Assign OSPF areas.
- Assign router interfaces to an OSPF area.

The following configuration tasks are optional:

- Configure a virtual link between an ABR without a physical connection to a backbone area and the Foundry device in the same area with a physical connection to the backbone area
- Change the reference bandwidth for the cost on OSPFv3 interfaces
- Configure the redistribution of routes into OSPFv3
- Configure default route origination
- Modify the shortest path first (SPF) timers
- Modify the administrative distances for OSPFv3 routes.
- Configure the OSPFv3 LSA pacing interval
- Modify how often the Foundry device checks on the elimination of the database overflow condition
- Modify the external link state database limit
- Modify the default values of OSPFv3 parameters for router interfaces
- Disable or reenable OSPFv3 event logging

## Enabling OSPFv3

Before enabling the Foundry device to run OSPFv3, you must do the following:

- Enable the forwarding of IPv6 traffic on the Foundry device using the **ipv6 unicast-routing** command.
- Enable IPv6 on each interface over which you plan to enable OSPFv3. You enable IPv6 on an interface by configuring an IPv6 address or explicitly enabling IPv6 on that interface.

For more information about performing these configuration tasks, see “Configuring Basic IPv6 Connectivity” on page 3-1.

By default, OSPFv3 is disabled. To enable OSPFv3, you must enable it globally.

To enable OSPFv3 globally, enter the following command:

```
BigIron(config-ospf-router)#ipv6 router ospf
BigIron(config-ospf6-router)#
```

After you enter this command, the Foundry device enters the IPv6 OSPF configuration level, where you can access several commands that allow you to configure OSPFv3.

***Syntax:*** [no] ipv6 router ospf

To disable OSPFv3, enter the **no** form of this command. If you disable OSPFv3, the Foundry device removes all the configuration information for the disabled protocol from the running-config. Moreover, when you save the configuration to the startup-config file after disabling one of these protocols, all the configuration information for the disabled protocol is removed from the startup-config file.

The CLI displays a warning message such as the following:

```
BigIron(config-ospf6-router)# no ipv6 router ospf
ipv6 router ospf mode now disabled. All ospf config data will be lost when writing
to flash!
```

If you have disabled the protocol but have not yet saved the configuration to the startup-config file and reloaded the software, you can restore the configuration information by re-entering the command to enable the protocol (for example, **ipv6 router ospf**). If you have already saved the configuration to the startup-config file and reloaded the software, the information is gone. If you are testing an OSPF configuration and are likely to disable and re-enable the protocol, you might want to make a backup copy of the startup-config file containing the protocol's configuration information. This way, if you remove the configuration information by saving the configuration after disabling the protocol, you can restore the configuration by copying the backup copy of the startup-config file onto the flash memory.

## Assigning OSPFv3 Areas

After OSPFv3 is enabled, you can assign OSPFv3 areas. You can assign an IPv4 address or a number as the ***area ID*** for each area. The area ID is representative of all IPv6 addresses (subnets) on a router interface. Each router interface can support one area.

An area can be **normal** or a **stub**.

- Normal – OSPF routers within a normal area can send and receive External Link State Advertisements (LSAs).
- Stub – OSPF routers within a stub area cannot send or receive External LSAs. In addition, OSPF routers in a stub area must use a default route to the area's Area Border Router (ABR) or Autonomous System Boundary Router (ASBR) to send traffic out of the area.

For example, to set up OSPFv3 areas 0.0.0.0, 200.5.0.0, 192.5.1.0, and 195.5.0.0, enter the following commands:

```
BigIron(config-ospf6-router)# area 0.0.0.0
BigIron(config-ospf6-router)# area 200.5.0.0
BigIron(config-ospf6-router)# area 192.5.1.0
BigIron(config-ospf6-router)# area 195.5.0.0
```

***Syntax:*** [no] area <number> | <ipv4-address>

The <number> | <ipv4-address> parameter specifies the area number, which can be a number or in IPv4 address format. If you specify a number, the number can be from 0 – 2,147,483,647.

**NOTE:** You can assign one area on a router interface.

### Assigning a Totally Stubby Area

By default, the Foundry device sends summary LSAs (LSA type 3) into stub areas. You can further reduce the number of LSAs sent into a stub area by configuring the Foundry device to stop sending summary LSAs into the area. You can disable the summary LSAs when you are configuring the stub area or later after you have configured the area.

This feature disables origination of summary LSAs into a stub area, but the Foundry device still accepts summary LSAs from OSPF neighbors and floods them to other areas. The Foundry device can form adjacencies with other routers regardless of whether summarization is enabled or disabled for areas on each router.

When you disable the summary LSAs, the change takes effect immediately. If you apply the option to a previously configured area, the router flushes all of the summary LSAs it has generated (as an ABR) from the area.

**NOTE:** This feature applies only when the Foundry device is configured as an Area Border Router (ABR) for the area. To completely prevent summary LSAs from being sent to the area, disable the summary LSAs on each OSPF router that is an ABR for the area.

For example, to disable summary LSAs for stub area 40 and specify an additional metric of 99, enter the following command:

```
BigIron(config-ospf6-router)# area 40 stub 99 no-summary
```

***Syntax:*** area <number> | <ipv4-address> stub <metric> [no-summary]

The <number> | <ipv4-address> parameter specifies the area number, which can be a number or in IPv4 address format. If you specify a number, the number can be from 0 – 2,147,483,647.

The **stub** <metric> parameter specifies an additional cost for using a route to or from this area and can be from 1 – 16777215. There is no default. Normal areas do not use the cost parameter.

The **no-summary** parameter applies only to stub areas and disables summary LSAs from being sent to the area.

### Assigning Interfaces to an Area

After you define OSPFv3 areas, you must assign router interfaces to the areas. All router interfaces must be assigned to one of the defined areas on an OSPF router. When an interface is assigned to an area, all corresponding subnets on that interface are automatically included in the assignment.

For example, to assign Ethernet interface 3/1 to area 192.5.0.0, enter the following commands:

```
BigIron(config)# interface Ethernet 3/1
BigIron(config-if-e100-3/1)# ipv6 ospf area 195.5.0.0
```

***Syntax:*** [no] ipv6 ospf area <number> | <ipv4-address>

The <number> | <ipv4-address> parameter specifies the area number, which can be a number or in IPv4 address format. If you specify a number, the number can be from 0 – 2,147,483,647.

To remove the interface from the specified area, use the **no** form of this command.

## Configuring Virtual Links

All ABRs must have either a direct or indirect link to an OSPF backbone area (0.0.0.0 or 0). If an ABR does not have a physical link to a backbone area, you can configure a virtual link from the ABR to another router within the same area that has a physical connection to the backbone area.

The path for a virtual link is through an area shared by the neighbor ABR (router with a physical backbone connection) and the ABR requiring a logical connection to the backbone.

Two parameters must be defined for all virtual links—transit area ID and neighbor router.

- The transit area ID represents the shared area of the two ABRs and serves as the connection point between the two routers. This number should match the area ID value.
- When assigned from the router interface requiring a logical connection, the neighbor router field is the router ID (IPv4 address) of the router that is physically connected to the backbone. When assigned from the router interface with the physical connection, the neighbor router is the router ID (IPv4) address of the router requiring a logical connection to the backbone.

**NOTE:** By default, the Foundry router ID is the IPv4 address configured on the lowest numbered loopback interface. If the Foundry device does not have a loopback interface, the default router ID is the lowest numbered IPv4 address configured on the device.

**NOTE:** When you establish an area virtual link, you must configure it on both of the routers (both ends of the virtual link).

For example, imagine that ABR1 in areas 1 and 2 is cut off from the backbone area (area 0). To provide backbone access to ABR1, you can add a virtual link between ABR1 and ABR2 in area 1 using area 1 as a transit area. To configure the virtual link, you define the link on the router that is at each end of the link. No configuration for the virtual link is required on the routers in the transit area.

To define the virtual link on ABR1, enter the following command on ABR1:

```
BigIron(config-ospf6-router)# area 1 virtual-link 209.157.22.1
```

To define the virtual link on ABR2, enter the following command on ABR2:

```
BigIron(config-ospf6-router)# area 1 virtual-link 10.0.0.1
```

***Syntax:*** area <number> | <ipv4-address> virtual-link <router-id>

The **area** <number> | <ipv4-address> parameter specifies the transit area.

The <router-id> parameter specifies the router ID of the OSPF router at the remote end of the virtual link. To display the router ID on a router, enter the **show ip** command.

### Assigning a Virtual Link Source Address

When routers at both ends of a virtual link need to communicate with one another, the source address included in the packets must be a global IPv6 address. Therefore, you must determine the global IPv6 address to be used by the routers for communication across the virtual link. You can specify that a router uses the IPv6 global address assigned to one of its interfaces.

For example, to specify the global IPv6 address assigned to Ethernet interface 3/1 on ABR1 as the source address for the virtual link on ABR1, enter the following command on ABR1:

```
BigIron(config-ospf6-router)# virtual-link-if-address interface ethernet 3/1
```

To specify the global IPv6 address assigned to tunnel interface 1 on ABR2 as the source address for the virtual link on ABR2, enter the following command on ABR2:

```
BigIron(config-ospf6-router)# virtual-link-if-address interface tunnel 1
```

***Syntax:*** virtual-link-if-address interface ethernet <port> | loopback <number> | tunnel <number> | ve <number>

The **ethernet | loopback | tunnel | ve** parameter specifies the interface from which the router derives the source IPv6 address for communication across the virtual link. If you specify an Ethernet interface, also specify the port number associated with the interface. If you specify a loopback, tunnel, or VE interface, also specify the number associated with the respective interface.

To delete the source address for the virtual link, use the **no** form of this command.

### Modifying Virtual Link Parameters

You can modify the following virtual link parameters:

- Dead-interval: The number of seconds that a neighbor router waits for a hello packet from the current router before declaring the router is down. The range is 1 – 65535 seconds. The default is 40 seconds.
- Hello-interval: The length of time between the transmission of hello packets. The range is 1 – 65535 seconds. The default is 10 seconds.
- Retransmit-interval: The interval between the re-transmission of link state advertisements to router adjacencies for this interface. The range is 0 – 3600 seconds. The default is 5 seconds.
- Transmit-delay: The period of time it takes to transmit Link State Update packets on the interface. The range is 0 – 3600 seconds. The default is 1 second.

**NOTE:** The values of the **dead-interval** and **hello-interval** parameters must be the same at both ends of a virtual link. Therefore, if you modify the values of these parameters at one end of a virtual link, you must remember to make the same modifications on the other end of the link.

The values of the other virtual link parameters do not require synchronization.

For example, to change the dead interval to 60 seconds on the virtual links defined on ABR1 and ABR2, enter the following command on ABR1:

```
BigIron(config-ospf6-router)# area 1 virtual-link 209.157.22.1
dead-interval 60
```

Enter the following command on ABR2:

```
BigIron(config-ospf6-router)# area 1 virtual-link 10.0.0.1 dead-interval 60
```

***Syntax:*** area <number> | <ipv4-address> virtual-link <router-id> [dead-interval <seconds> | hello-interval <seconds> | retransmit-interval <seconds> | transmit-delay <seconds>]

The **area** <number> | <ipv4-address> parameter specifies the transit area.

The <router-id> parameter specifies the router ID of the OSPF router at the remote end of the virtual link. To display the router ID on a router, enter the **show ip** command.

The **dead-interval**, **hello-interval**, **retransmit-interval**, and **transmit-delay** parameters are discussed earlier in this section.

## Changing the Reference Bandwidth for the Cost on OSPFv3 Interfaces

Each interface on which OSPFv3 is enabled has a cost associated with it. The Foundry device advertises its interfaces and their costs to OSPFv3 neighbors. For example, if an interface has an OSPF cost of ten, the Foundry device advertises the interface with a cost of ten to other OSPF routers.

By default, an interface's OSPF cost is based on the port speed of the interface. The software uses the following formula to calculate the cost:

Cost = reference-bandwidth/interface-speed

By default, the reference bandwidth is 100 Mbps. If the resulting cost is less than 1, the software rounds the cost up to 1. The default reference bandwidth results in the following costs:

- 10 Mbps port's cost = 100/10 = 10
- 100 Mbps port's cost = 100/100 = 1
- 1000 Mbps port's cost = 100/1000 = 0.10, which is rounded up to 1
- 155 Mbps port's cost = 100/155 = 0.65, which is rounded up to 1
- 622 Mbps port's cost = 100/622 = 0.16, which is rounded up to 1
- 2488 Mbps port's cost = 100/2488 = 0.04, which is rounded up to 1

The bandwidth for interfaces that consist of more than one physical port is calculated as follows:

- Trunk group – The combined bandwidth of all the ports.
- Virtual (Ethernet) interface – The combined bandwidth of all the ports in the port-based VLAN that contains the virtual interface.

You can change the default reference bandwidth from 100 Mbps to a value from 1 – 4294967 Mbps.

If a change to the reference bandwidth results in a cost change to an interface, the Foundry device sends a link state update to update the costs of interfaces advertised by the Foundry device.

**NOTE:** If you specify the cost for an individual interface, the cost you specify overrides the cost calculated by the software.

Some interface types are not affected by the reference bandwidth and always have the same cost regardless of the reference bandwidth in use:

- The cost of a loopback interface is always 0.
- The cost of a virtual link is calculated using the Shortest Path First (SPF) algorithm and is not affected by the auto-cost feature.

- The bandwidth for tunnel interfaces is 9 Kbps and is not affected by the auto-cost feature.

For example, to change the reference bandwidth to 500, enter the following command:

```
BigIron(config-ospf6-router)# auto-cost reference-bandwidth 500
```

The reference bandwidth specified in this example results in the following costs:

- 10 Mbps port's cost = 500/10 = 50
- 100 Mbps port's cost = 500/100 = 5
- 1000 Mbps port's cost = 500/1000 = 0.5, which is rounded up to 1
- 155 Mbps port's cost = 500/155 = 3.23, which is rounded up to 4
- 622 Mbps port's cost = 500/622 = 0.80, which is rounded up to 1
- 2488 Mbps port's cost = 500/2488 = 0.20, which is rounded up to 1

The costs for 10 Mbps, 100 Mbps, and 155 Mbps ports change as a result of the changed reference bandwidth. Costs for higher-speed interfaces remain the same.

***Syntax:*** [no] auto-cost reference-bandwidth <number>

The <number> parameter specifies the reference bandwidth and can be a value from 1 – 4294967. The default is 100.

To restore the reference bandwidth to its default value and thus restore the default costs of interfaces to their default values, enter the **no** form of this command.

## Redistributing Routes into OSPFv3

In addition to specifying which routes are redistributed into OSPFv3, you can configure the following aspects related to route redistribution:

- Default metric.
- Metric type.
- Advertisement of an external aggregate route.

### Configuring Route Redistribution into OSPFv3

You can configure the Foundry device to redistribute routes from the following sources into OSPFv3:

- IPv6 static routes.
- Directly connected IPv6 networks.
- BGP4+.
- IPv6 IS-IS.
- RIPng.

You can redistribute routes in the following ways:

- By route types, for example, the Foundry device redistributes all IPv6 static and RIPng routes.
- By using a route map to filter which routes to redistribute, for example, the Foundry device redistributes specified IPv6 static and RIPng routes only.

For example, to configure the redistribution of all IPv6 static, RIPng, and IPv6 IS-IS level-1 and level-2 routes, enter the following commands:

```
BigIron(config-ospf6-router)# redistribute static
BigIron(config-ospf6-router)# redistribute rip
BigIron(config-ospf6-router)# redistribute isis level-1-2
```

***Syntax:*** [no] redistribute bgp | connected | isis [level-1 | level-1-2 | level-2] | rip | static [metric <number> | metric-type <type>]

The **bgp** | **connected** | **isis** | **rip** | **static** keywords specify the route source.

The **level-1** | **level-1-2** | **level-2** keywords (for IPv6 IS-IS only) allow you to specify that the Foundry device redistributes level-1 routes only, level-2 routes only, or both level-1 and level-2 routes.

The **metric** <number> parameter specifies the metric used for the redistributed route. If a value is not specified for this option, and the value for the **default-metric** command is set to 0, its default metric, then routes redistributed from the various routing protocols will have the metric value of the protocol from which they are redistributed. For information about the **default-metric** command, see “Modifying Default Metric for Routes Redistributed into OSPF Version 3” on page 6-9.

The **metric-type** <type> parameter specifies an OSPF metric type for the redistributed route. You can specify external type 1 or external type 2. If a value is not specified for this option, the Foundry device uses the value specified by the **metric-type** command. For information about modifying the default metric type using the **metric-type** command, see “Modifying Metric Type for Routes Redistributed into OSPF Version 3” on page 6-9.

For example, to configure a route map and use it for redistribution of routes into OSPFv3, enter commands such as the following:

```
BigIron(config)# ipv6 route 2001:1::/32 4823:eoff:343e::23
BigIron(config)# ipv6 route 2001:2::/32 4823:eoff:343e::23
BigIron(config)# ipv6 route 2001:3::/32 4823:eoff:343e::23 metric 5
BigIron(config)# route-map abc permit 1
BigIron(config-routemap abc)# match metric 5
BigIron(config-routemap abc)# set metric 8
BigIron(config-routemap abc)# ipv6 router ospf
BigIron(config-ospf6-router)# redistribute static route-map abc
```

The commands in this example configure some static IPv6 routes and a route map, and use the route map for redistributing the static IPv6 routes into OSPFv3.

The **ipv6 route** commands configure the static IPv6 routes. The **route-map** command begins configuration of a route map called “abc”. The number indicates the route map entry (called the “instance”) you are configuring. A route map can contain multiple entries. The software compares packets to the route map entries in ascending numerical order and stops the comparison once a match is found.

The **match** command in the route map matches on routes that have 5 for their metric value (cost). The **set** command changes the metric in routes that match the route map to 8.

The **redistribute** command configures the redistribution of static IPv6 routes into OSPFv3, and uses route map “abc“ to control the routes that are redistributed. In this example, the route map allows a static IPv6 route to be redistributed into OSPF only if the route has a metric of 5, and changes the metric to 8 before placing the route into the OSPF route redistribution table.

***Syntax:*** [no] redistribute bgp | connected | isis | rip | static [route-map <map-name>]

The **bgp** | **connected** | **isis** | **rip** | **static** keywords specify the route source.

The **route-map** <map-name> parameter specifies the route map name. The following match parameters are valid for OSPFv3 redistribution:

- **match ip address** | **next-hop** <acl-number>
- **match metric** <number>
- **match tag** <tag-value>

The following set parameters are valid for OSPF redistribution:

- **set ip next hop** <ipv4-address>
- **set metric** [+ | - ] <number> | none
- **set metric-type** type-1 | type-2
- **set tag** <tag-value>

**NOTE:** You must configure the route map before you configure a redistribution filter that uses the route map.

**NOTE:** When you use a route map for route redistribution, the software disregards the permit or deny action of the route map.

**NOTE:** For an external route that is redistributed into OSPFv3 through a route map, the metric value of the route remains the same unless the metric is set by a **set metric** command inside the route map or the **default-metric** <num> command. For a route redistributed without using a route map, the metric is set by the metric parameter if set or the **default-metric** <num> command if the metric parameter is not set.

### Modifying Default Metric for Routes Redistributed into OSPF Version 3

The default metric is a global parameter that specifies the cost applied by default to routes redistributed into OSPFv3. The default value is 0.

If the **metric** parameter for the **redistribute** command is not set and the **default-metric** command is set to 0, its default value, then routes redistributed from the various routing protocols will have the metric value of the protocol from which they are redistributed. For information about the **redistribute** command, see “Configuring Route Redistribution into OSPFv3” on page 6-7.

**NOTE:** You also can define the cost on individual interfaces. The interface cost overrides the default cost. For information about defining the cost on individual interfaces, see “Modifying OSPFv3 Interface Defaults” on page 6-15 and “Changing the Reference Bandwidth for the Cost on OSPFv3 Interfaces” on page 6-6.

To assign a default metric of 4 to all routes imported into OSPFv3, enter the following command:

```
BigIron(config-ospf6-router)# default-metric 4
```

***Syntax:*** [no] default-metric <number>

You can specify a value from 0 – 65535. The default is 0.

To restore the default metric to the default value, use the **no** form of this command.

### Modifying Metric Type for Routes Redistributed into OSPF Version 3

The Foundry device uses the **metric-type** parameter by default for all routes redistributed into OSPFv3 unless you specify a different metric type for individual routes using the **redistribute** command. (For more information about using the **redistribute** command, see “Redistributing Routes into OSPFv3” on page 6-7.)

A type 1 route specifies a small metric (two bytes), while a type 2 route specifies a big metric (three bytes). The default value is type 2.

To modify the default value of type 2 to type 1, enter the following command:

```
BigIron(config-ospf6-router)# metric-type type1
```

***Syntax:*** [no] metric-type type1 | type2

To restore the metric type to the default value, use the **no** form of this command.

### Configuring External Route Summarization

When the Foundry device is an OSPF Autonomous System Boundary Router (ASBR), you can configure it to advertise one external route as an aggregate for all redistributed routes that are covered by a specified IPv6 address range.

When you configure an address range, the range takes effect immediately. All the imported routes are summarized according to the configured address range. Imported routes that have already been advertised and that fall within the range are flushed out of the AS and a single route corresponding to the range is advertised.

If a route that falls within a configured address range is imported by the Foundry device, no action is taken if the device has already advertised the aggregate route; otherwise, the device advertises the aggregate route. If an

imported route that falls within a configured address range is removed by the device, no action is taken if there are other imported route(s) that fall with in the same address range; otherwise the aggregate route is flushed.

You can configure up to 32 address ranges. The Foundry device sets the forwarding address of the aggregate route to zero and sets the tag to zero.

If you delete an address range, the advertised aggregate route is flushed and all imported routes that fall within the range are advertised individually.

If an external link state database overflow (LSDB) condition occurs, all aggregate routes are flushed out of the AS, along with other external routes. When the Foundry device exits the external LSDB overflow condition, all the imported routes are summarized according to the configured address ranges.

**NOTE:** If you use redistribution filters in addition to address ranges, the Foundry device applies the redistribution filters to routes first, then applies them to the address ranges.

**NOTE:** If you disable redistribution, all the aggregate routes are flushed, along with other imported routes.

**NOTE:** This option affects only imported, type 5 external routes. A single type 5 LSA is generated and flooded throughout the AS for multiple external routes.

To configure the summary address 2201::/24 for routes redistributed into OSPFv3, enter the following command:

```
BigIron(config-ospf6-router)# summary-address 2201::/24
```

In this example, the summary prefix 2201::/24 includes addresses 2201::/1 through 2201::/24. Only the address FEC0::/24 is advertised in an external link-state advertisement.

***Syntax:*** summary-address <ipv6-prefix>/<prefix-length>

You must specify the <ipv6-prefix> parameter in hexadecimal using 16-bit values between colons as documented in RFC 2373.

You must specify the <prefix-length> parameter as a decimal value. A slash mark (/) must follow the <ipv6-prefix> parameter and precede the <prefix-length> parameter.

## Filtering OSPFv3 Routes

You can filter the routes to be placed in the OSPFv3 route table by configuring distribution lists. OSPFv3 distribution lists can be applied globally or to an interface.

The functionality of OSPFv3 distribution lists is similar to that of OSPFv2 distribution lists. However, unlike OSPFv2 distribution lists, which filter routes based on criteria specified in an Access Control List (ACL), OSPFv3 distribution lists can filter routes using information specified in an IPv6 prefix list or a route map.

### Configuration Examples

The following sections show examples of filtering OSPFv3 routes using prefix lists globally and for a specific interface, as well as filtering OSPFv3 routes using a route map.

You can configure the device to use all three types of filtering. When you do this, filtering using route maps has higher priority over filtering using global prefix lists. Filtering using prefix lists for a specific interface has lower priority than the other two filtering methods.

The example in this section assume the following routes are in the OSPFv3 route table:

```
BigIron# show ipv6 ospf route

  Current Route count: 5
    Intra: 3 Inter: 0 External: 2 (Type1 0/Type2 2)
    Equal-cost multi-path: 0
    Destination                    Options   Area            Cost Type2 Cost
    Next Hop Router                Outgoing Interface
*IA 3001::/64                      --------- 0.0.0.1            0 0
    ::                             ve 10
*E2 3010::/64                      --------- 0.0.0.0           10 0
    fe80::2e0:52ff:fe00:10         ve 10
*IA 3015::/64                      V6E---R-- 0.0.0.0           11 0
    fe80::2e0:52ff:fe00:10         ve 10
*IA 3020::/64                      --------- 0.0.0.0           10 0
    ::                             ve 11
*E2 6001:5000::/64                 --------- 0.0.0.0           10 0
    fe80::2e0:52ff:fe00:10         ve 10
```

### *Configuring an OSPFv3 Distribution List Using an IPv6 Prefix List as Input*

The following example illustrates how to use an IPv6 prefix list is used to filter OSPFv3 routes.

To specify an IPv6 prefix list called filterOspfRoutes that denies route 3010::/64, enter the following commands:

```
BigIron(config)# ipv6 prefix-list  filterOspfRoutes seq 5 deny 3010::/64
BigIron(config)# ipv6 prefix-list  filterOspfRoutes seq 7 permit ::/0 ge 1 le 128
```

***Syntax:*** ipv6 prefix-list <name> [seq <seq-value>] [description <string>] deny | permit <ipv6-addr>/<mask-bits> [ge <ge-value>] [le <le-value>]

See "Defining IP Prefix Lists" in the *Foundry Router Configuration Guide* for information on configuring prefix lists.

To configure a distribution list that applies the filterOspfRoutes prefix list globally:

```
BigIron(config)# ipv6 router ospf
BigIron(config-ospf6-router)# distribute-list prefix-list filterOspfRoutes in
```

***Syntax:*** [no] distribute-list prefix-list <name> in [<interface>]

After this distribution list is configured, route 3010::/64 would be omitted from the OSPFv3 route table:

```
BigIron# show ipv6 ospf route

  Current Route count: 4
    Intra: 3 Inter: 0 External: 1 (Type1 0/Type2 1)
    Equal-cost multi-path: 0
    Destination                    Options   Area            Cost Type2 Cost
    Next Hop Router                Outgoing Interface
*IA 3001::/64                      --------- 0.0.0.1            0 0
    ::                             ve 10
*IA 3015::/64                      V6E---R-- 0.0.0.0           11 0
    fe80::2e0:52ff:fe00:10         ve 10
*IA 3020::/64                      --------- 0.0.0.0           10 0
    ::                             ve 11
*E2 6001:5000::/64                 --------- 0.0.0.0           10 0
    fe80::2e0:52ff:fe00:10         ve 10
```

The following commands specify an IPv6 prefix list called filterOspfRoutesVe that denies route 3015::/64:

```
BigIron(config)# ipv6 prefix-list filterOspfRoutesVe seq 5 deny 3015::/64
BigIron(config)# ipv6 prefix-list filterOspfRoutesVe seq 10 permit ::/0 ge 1 le 128
```

The following commands configure a distribution list that applies the filterOspfRoutesVe prefix list to routes pointing to virtual interface 10:

```
BigIron(config)# ipv6 router ospf
BigIron(config-ospf6-router)# distribute-list prefix-list filterOspfRoutes in ve 10
```

After this list is configured, route 3015::/64, pointing to virtual interface 10, is omitted from the OSPFv3 route table:

```
BigIron# show ipv6 ospf route

  Current Route count: 4
    Intra: 3 Inter: 0 External: 1 (Type1 0/Type2 1)
    Equal-cost multi-path: 0
    Destination                    Options   Area            Cost Type2 Cost
    Next Hop Router                Outgoing Interface
*IA 3001::/64                      --------- 0.0.0.1           0 0
    ::                             ve 10
*E2 3010::/64                      --------- 0.0.0.0          10 0
    fe80::2e0:52ff:fe00:10         ve 10
*IA 3020::/64                      --------- 0.0.0.0          10 0
    ::                             ve 11
*E2 6001:5000::/64                 --------- 0.0.0.0          10 0
    fe80::2e0:52ff:fe00:10         ve 10
```

### *Configuring an OSPFv3 Distribution List Using a Route Map as Input*

The following commands configure a route map that matches internal routes:

```
BigIron(config)# route-map allowInternalRoutes permit 10
BigIron(config-routemap allowInternalRoutes)# match route-type internal
```

See "Hardware-Based Policy-Based Routing (PBR)" in the *Foundry Enterprise Configuration and Management Guide* for information on configuring route maps.

The following commands configure a distribution list that applies the allowInternalRoutes route map globally to OSPFv3 routes:

```
BigIron(config)# ipv6 router ospf
BigIron(config-ospf6-router)# distribute-list route-map allowinternalroutes in
```

***Syntax:*** [no] distribute-list route-map <name> in

After this distribution list is configured, the internal routes would be included, and the external routes would be omitted from the OSPFv3 route table:

```
BigIron# show ipv6 ospf route

  Current Route count: 3
    Intra: 3 Inter: 0 External: 0 (Type1 0/Type2 0)
    Equal-cost multi-path: 0
    Destination                    Options   Area            Cost Type2 Cost
    Next Hop Router                Outgoing Interface
*IA 3001::/64                      --------- 0.0.0.1           0 0
    ::                             ve 10
*IA 3015::/64                      V6E---R-- 0.0.0.0          11 0
    fe80::2e0:52ff:fe00:10         ve 10
*IA 3020::/64                      --------- 0.0.0.0          10 0
    ::                             ve 11
```

## Configuring Default Route Origination

When the Foundry device is an OSPF Autonomous System Boundary Router (ASBR), you can configure it to automatically generate a default external route into an OSPFv3 routing domain. This feature is called “default route origination” or “default information origination.”

By default, the Foundry device does not advertise the default route into the OSPFv3 domain. If you want the device to advertise the OSPF default route, you must explicitly enable default route origination.

When you enable OSPF default route origination, the device advertises a type 5 default route that is flooded throughout the AS (except stub areas).

The device advertises the default route into OSPF even if OSPF route redistribution is not enabled, and even if the default route is learned through an IBGP neighbor.

**NOTE:** The Foundry device does not advertise the OSPF default route, regardless of other configuration parameters, unless you explicitly enable default route origination.

If default route origination is enabled and you disable it, the default route originated by the device is flushed. Default routes generated by other OSPF routers are not affected. If you re-enable the feature, the feature takes effect immediately and thus does not require you to reload the software.

For example, to create and advertise a default route with a metric of 2 and as a type 1 external route, enter the following command:

```
BigIron(config-ospf6-router)# default-information-originate always metric 2 metric-type type1
```

***Syntax:*** [no] default-information-originate [always] [metric <value>] [metric-type <type>]

The **always** keyword originates a default route regardless of whether the device has learned a default route. This option is disabled by default.

The **metric** <value> parameter specifies a metric for the default route. If this option is not used, the value of the **default-metric** command is used for the route. For information about this command, see “Modifying Default Metric for Routes Redistributed into OSPF Version 3” on page 6-9.

The **metric-type** <type> parameter specifies the external link type associated with the default route advertised into the OSPF routing domain. The <type> can be one of the following:

- 1 – Type 1 external route
- 2 – Type 2 external route

If you do not use this option, the default redistribution metric type is used for the route type.

**NOTE:** If you specify a metric and metric type, your values are used even if you do not use the always option.

To disable default route origination, enter the **no** form of the command.

## Modifying Shortest Path First Timers

The Foundry device uses the following timers when calculating the shortest path for OSPFv3 routes:

- SPF delay – When the Foundry device receives a topology change, the software waits before it starts a Shortest Path First (SPF) calculation. By default, the software waits 5 seconds. You can configure the SPF delay to a value from 0 – 65535 seconds. If you set the SPF delay to 0 seconds, the software immediately begins the SPF calculation after receiving a topology change.
- SPF hold time – The Foundry device waits a specific amount of time between consecutive SPF calculations. By default, the device waits 10 seconds. You can configure the SPF hold time to a value from 0 – 65535 seconds. If you set the SPF hold time to 0 seconds, the software does not wait between consecutive SPF calculations.

You can set the SPF delay and hold time to lower values to cause the device to change to alternate paths more quickly if a route fails. Note that lower values for these parameters require more CPU processing time.

You can change one or both of the timers.

**NOTE:** If you want to change only one of the timers, for example, the SPF delay timer, you must specify the new value for this timer as well as the current value of the SPF hold timer, which you want to retain. The Foundry device does not accept only one timer value.

To change the SPF delay to 10 seconds and the SPF hold to 20 seconds, enter the following command:

```
BigIron(config-ospf6-router)# timers spf 10 20
```

***Syntax:*** timers spf <delay> <hold-time>

For the <delay> and <hold-time> parameters, specify a value from 0 – 65535 seconds.

To set the timers back to their default values, enter the **no** version of this command.

## Modifying Administrative Distance

The Foundry device can learn about networks from various protocols, including BGP4+, IPv6 IS-IS, RIPng, and OSPFv3. Consequently, the routes to a network may differ depending on the protocol from which the routes were learned. By default, the administrative distance for OSPFv3 routes is 110.

The device selects one route over another based on the source of the route information. To do so, the device can use the administrative distances assigned to the sources. You can influence the device's decision by changing the default administrative distance for OSPFv3 routes.

### Configuring Administrative Distance Based on Route Type

You can configure a unique administrative distance for each type of OSPFv3 route. For example, you can use this feature to influence the Foundry device to prefer a static route over an OSPF inter-area route and to prefer OSPF intra-area routes to static routes.

The distance you specify influences the choice of routes when the device has multiple routes to the same network from different protocols. The device prefers the route with the lower administrative distance.

You can specify unique default administrative distances for the following OSPFv3 route types:

- Intra-area routes
- Inter-area routes
- External routes

The default for all of these OSPFv3 route types is 110.

**NOTE:** This feature does not influence the choice of routes within OSPFv3. For example, an OSPF intra-area route is always preferred over an OSPF inter-area route, even if the intra-area route's distance is greater than the inter-area route's distance.

For example, to change the default administrative distances for intra-area routes to 80, inter-area routes to 90, and external routes to 100, enter the following commands:

```
BigIron(config-ospf6-router)# distance intra-area 80
BigIron(config-ospf6-router)# distance inter-area 90
BigIron(config-ospf6-router)# distance external 100
```

***Syntax:*** distance external | inter-area | intra-area <distance>

The **external** | **inter-area** | **intra-area** keywords specify the route type for which you are changing the default administrative distance.

The <distance> parameter specifies the new distance for the specified route type. Specify a value from 1 – 255.

To reset the administrative distance of a route type to its system default, enter the **no** form of this command.

## Configuring the OSPFv3 LSA Pacing Interval

The Foundry device paces OSPFv3 LSA refreshes by delaying the refreshes for a specified time interval instead of performing a refresh each time an individual LSA's refresh timer expires. The accumulated LSAs constitute a group, which the Foundry device refreshes and sends out together in one or more packets.

The pacing interval, which is the interval at which the Foundry device refreshes an accumulated group of LSAs, is configurable to a range from 10 – 1800 seconds (30 minutes). The default is 240 seconds (four minutes). Thus, every four minutes, the Foundry device refreshes the group of accumulated LSAs and sends the group together in the same packet(s).

The pacing interval is inversely proportional to the number of LSAs the Foundry device is refreshing and aging. For example, if you have approximately 10,000 LSAs, decreasing the pacing interval enhances performance. If you have a very small database (40 – 100 LSAs), increasing the pacing interval to 10 – 20 minutes might enhance performance only slightly.

To change the OSPFv3 LSA pacing interval to two minutes (120 seconds), enter the following command:

```
BigIron(config)# ipv6 router ospf
BigIron(config-ospf6-router)# timers lsa-group-pacing 120
```

***Syntax:*** [no] timers lsa-group-pacing <seconds>

The <seconds> parameter specifies the number of seconds and can be from 10 – 1800 (30 minutes). The default is 240 seconds (four minutes).

To restore the pacing interval to its default value, use the **no** form of the command:

## Modifying Exit Overflow Interval

If a database overflow condition occurs on the Foundry device, the device eliminates the condition by removing entries that originated on the device. The exit overflow interval allows you to set how often a device checks to see if the overflow condition has been eliminated. The default value is 0. If the configured value of the database overflow interval is 0, then the device never leaves the database overflow condition.

For example, to modify the exit overflow interval to 60 seconds, enter the following command:

```
BigIron(config-ospf6-router)# database-overflow-interval 60
```

***Syntax:*** database-overflow-interval <seconds>

The <seconds> parameter can be a value from 0 – 86400 seconds (24 hours).

To reset the exit overflow interval to its system default, enter the **no** form of this command.

## Modifying External Link State Database Limit

By default, the link state database can hold a maximum of 2000 entries for external (type 5) LSAs. You can change the maximum number of entries from 500 – 8000. After changing this limit, make sure to save the running-config file and reload the software. The change does not take effect until you reload or reboot the software.

For example, to change the maximum number entries from the default of 2000 to 3000, enter the following command:

```
BigIron(config-ospf6-router)# external-lsdb-limit 3000
```

***Syntax:*** external-lsdb-limit <entries>

The <entries> parameter can be a numerical value from 500 – 8000 seconds.

To reset the maximum number of entries to its system default, enter the **no** form of this command.

## Modifying OSPFv3 Interface Defaults

OSPFv3 has interface parameters that you can configure. For simplicity, each of these parameters has a default value. No change to these default values is required except as needed for specific network configurations.

You can modify the default values for the following OSPF interface parameters:

- Cost: Indicates the overhead required to send a packet across an interface. You can modify the cost to differentiate between 100 Mbps and 1000 Mbps (1 Gbps) links. The command syntax is **ipv6 ospf cost** <number>. The default cost is calculated by dividing 100 million by the bandwidth. For 10 Mbps links, the cost is 10. The cost for both 100 Mbps and 1000 Mbps links is 1, because the speed of 1000 Mbps was not in use at the time the OSPF cost formula was devised.
- Dead-interval: Indicates the number of seconds that a neighbor router waits for a hello packet from the current router before declaring the router down. The command syntax is **ipv6 ospf dead-interval** <seconds>. The value can be from 1 – 2147483647 seconds. The default is 40 seconds.
- Hello-interval: Represents the length of time between the transmission of hello packets. The command syntax is **ipv6 ospf hello-interval** <seconds>. The value can be from 1 – 65535 seconds. The default is 10 seconds.
- Instance: Indicates the number of OSPFv3 instances running on an interface. The command syntax is **ipv6 ospf instance** <number>. The value can be from 0 – 255. The default is 1.
- MTU-ignore: Allows you to disable a check that verifies the same MTU is used on an interface shared by neighbors. The command syntax is **ipv6 ospf mtu-ignore**. By default, the mismatch detection is enabled.
- Network: Allows you to configure the OSPF network type. The command syntax is **ipv6 ospf network** [**point-to-multipoint**]. The default setting of the parameter depends on the network type.
- Passive: When you configure an OSPF interface to be passive, that interface does not send or receive OSPF route updates. This option affects all IPv6 subnets configured on the interface. The command syntax is **ipv6 ospf passive**. By default, all OSPF interfaces are active and thus can send and receive OSPF route information. Since a passive interface does not send or receive route information, the interface is in effect a stub network.
- Priority: Allows you to modify the priority of an OSPF router. The priority is used when selecting the designated router (DR) and backup designated routers (BDRs). The command syntax is **ipv6 ospf priority** <number>. The value can be from 0 – 255. The default is 1. If you set the priority to 0, the router does not participate in DR and BDR election.
- Retransmit-interval: The time between retransmissions of LSAs to adjacent routers for an interface. The command syntax is **ipv6 ospf retransmit-interval** <seconds>. The value can be from 0 – 3600 seconds. The default is 5 seconds.
- Transmit-delay: The time it takes to transmit Link State Update packets on this interface. The command syntax is **ipv6 ospf transmit-delay** <seconds>. The value can be from 0 – 3600 seconds. The default is 1 second.

### Disabling or Reenabling Event Logging

OSPFv3 does not currently support the generation of SNMP traps. Instead, you can disable or reenable the logging of OSPFv3-related events such as neighbor state changes and database overflow conditions. By default, the Foundry device logs these events.

To disable the logging of events, enter the following command:

```
BigIron(config-ospf6-router)# no log-status-change
```

***Syntax:*** [no] log-status-change

To reenable the logging of events, enter the following command:

```
BigIron(config-ospf6-router)# log-status-change
```

## Displaying OSPFv3 Information

You can display the information for the following OSPFv3 parameters:

- Areas
- Link state databases
- Interfaces

- Memory usage
- Neighbors
- Redistributed routes
- Routes
- SPF
- Virtual links
- Virtual neighbors

## Displaying OSPFv3 Area Information

To display global OSPFv3 area information for the Foundry device, enter the following command at any CLI level:

```
BigIron# show ipv6 ospf area
Area 0:
  Interface attached to this area: loopback 2 ethe 3/2 tunnel 2
  Number of Area scoped LSAs is 6
  Statistics of Area 0:
    SPF algorithm executed 16 times
    SPF last updated: 335256 sec ago
    Current SPF node count: 3
      Router: 2 Network: 1
      Maximum of Hop count to nodes: 2
...
```

***Syntax:*** show ipv6 ospf area [<area-id>]

You can specify the <area-id> parameter in the following formats:

- As an IPv4 address, for example, 192.168.1.1.
- As a numerical value from 0 – 2,147,483,647.

The <area-id> parameter restricts the display to the specified OSPF area.

This display shows the following information:

**Table 6.1: OSPFv3 area information fields**

| This Field... | Displays... |
|---|---|
| Area | The area number. |
| Interface attached to this area | The router interfaces attached to the area. |
| Number of Area scoped LSAs | Number of LSAs with a scope of the specified area. |
| SPF algorithm executed | The number of times the OSPF Shortest Path First (SPF) algorithm is executed within the area. |
| SPF last updated | The interval in seconds that the SPF algorithm was last executed within the area. |
| Current SPF node count | The current number of SPF nodes in the area. |
| Router | Number of router LSAs in the area. |
| Network | Number of network LSAs in the area. |

**Table 6.1: OSPFv3 area information fields (Continued)**

| This Field... | Displays... |
|---|---|
| Indx | The row number of the entry in the router's OSPF area table. |
| Area | The area number. |
| Maximum hop count to nodes. | The maximum number of hop counts to an SPF node within the area. |

## Displaying OSPFv3 Database Information

You can display a summary of the Foundry device's link state database or detailed information about a specified LSA type.

To display a summary of a device's link state database, enter the following command at any CLI level:

```
BigIron# show ipv6 ospf database
Area ID         Type LS ID    Adv Rtr         Seq(Hex) Age  Cksum  Len
0               Link 000001e6 223.223.223.223 800000ab 1547 8955   68
0               Link 000000d8 1.1.1.1         800000aa 1295 0639   68
0               Link 00000185 223.223.223.223 800000ab 1481 7e6b   56
0               Iap  00000077 223.223.223.223 800000aa 1404 966a   56
0               Rtr  00000124 223.223.223.223 800000b0 1397 912c   40
0               Net  00000016 223.223.223.223 800000aa 1388 1b09   32
0               Iap  000001d1 223.223.223.223 800000bd 1379 a072   72
0               Iap  000000c3 1.1.1.1         800000ae 1325 e021   52
0               Rtr  00000170 1.1.1.1         800000ad 1280 af8e   40
N/A             Extn 00000062 223.223.223.223 800000ae 1409 0ca7   32
N/A             Extn 0000021d 223.223.223.223 800000a8 1319 441e   32
```

***Syntax:*** show ipv6 ospf database [advrtr <ipv4-address> | as-external | extensive | inter-prefix | inter-router | intra-prefix | link | link-id <number> | network | router [scope <area-id> | as | link]]

The **advrtr** <ipv4-address> parameter displays detailed information about the LSAs for a specified advertising router only.

The **as-external** keyword displays detailed information about the AS externals LSAs only.

The **extensive** keyword displays detailed information about all LSAs in the database.

The **inter-prefix** keyword displays detailed information about the inter-area prefix LSAs only.

The **inter-router** keyword displays detailed information about the inter-area router LSAs only.

The **intra-prefix** keyword displays detailed information about the intra-area prefix LSAs only.

The **link** keyword displays detailed information about the link LSAs only.

The **link-id** <number> parameter displays detailed information about the specified link LSAs only.

The **network** <number> displays detailed information about the network LSAs only.

The **router** <number> displays detailed information about the router LSAs only.

The **scope** <area-id> parameter displays detailed information about the LSAs for a specified area, AS, or link.

This display shows the following information:

**Table 6.2: OSPFv3 database summary fields**

| This Field... | Displays... |
|---|---|
| Area ID | The OSPF area in which the Foundry device resides. |
| Type | Type of LSA. LSA types can be the following:<br>• Rtr – Router LSAs (Type 1).<br>• Net – Network LSAs (Type 2).<br>• Inap – Inter-area prefix LSAs for ABRs (Type 3).<br>• Inar – Inter-area router LSAs for ASBRs (Type 4).<br>• Extn – AS external LSAs (Type 5).<br>• Link – Link LSAs (Type 8).<br>• Iap – Intra-area prefix LSAs (Type 9). |
| LS ID | The ID of the LSA, in hexadecimal, from which the device learned this route. |
| Adv Rtr | The device that advertised the route. |
| Seq(Hex) | The sequence number of the LSA. The OSPF neighbor that sent the LSA stamps it with a sequence number to enable the Foundry device and other OSPF routers to determine which LSA for a given route is the most recent. |
| Age | The age of the LSA, in seconds. |
| Chksum | A checksum for the LSA packet. The checksum is based on all the fields in the packet except the age field. The Foundry device uses the checksum to verify that the packet is not corrupted. |
| Len | The length, in bytes, of the LSA. |

For example, to display detailed information about all LSAs in the database, enter the following command at any CLI level:

```
BigIron# show ipv6 ospf database extensive
Area ID         Type LS ID    Adv Rtr         Seq(Hex) Age  Cksum  Len
0               Link 00000031 1.1.1.1         80000001 35   6db9   56
     Router Priority: 1
     Options: V6E---R--
     LinkLocal Address: fe80::1
     Number of Prefix: 1
     Prefix Options:
     Prefix: 3002::/64
    ...
Area ID         Type LS ID    Adv Rtr         Seq(Hex) Age  Cksum  Len
0               Iap  00000159 223.223.223.223 800000ab 357  946b   56
     Number of Prefix: 2
     Referenced LS Type: Network
     Referenced LS ID: 00000159
     Referenced Advertising Router: 223.223.223.223
     Prefix Options:  Metric: 0
     Prefix: 2000:4::/64
     Prefix Options:  Metric: 0
     Prefix: 2002:c0a8:46a::/64
Area ID         Type LS ID    Adv Rtr         Seq(Hex) Age  Cksum  Len
0               Rtr  00000039 223.223.223.223 800000b1 355  8f2d   40
  Capability Bits: --E-
  Options: V6E---R--
  Type: Transit Metric: 1
  Interface ID: 00000058  Neighbor Interface ID: 00000058
  Neighbor Router ID: 223.223.223.223
Area ID         Type LS ID    Adv Rtr         Seq(Hex) Age  Cksum  Len
0               Net  000001f4 223.223.223.223 800000ab 346  190a   32
     Options: V6E---R--
     Attached Router: 223.223.223.223
     Attached Router: 1.1.1.1
...
Area ID         Type LS ID    Adv Rtr         Seq(Hex) Age  Cksum  Len
N/A             Extn 000001df 223.223.223.223 800000af 368  0aa8   32
     Bits: E
     Metric: 00000001
     Prefix Options:
     Referenced LSType: 0
     Prefix: 2002::/16

Area ID         Type LS ID    Adv Rtr         Seq(Hex) Age  Cksum  Len
1               Inap 0000011d 10.1.1.188      80000001 124  25de   36
     Metric: 2
     Prefix Options:
     Prefix: 2000:2:2::/64

Area ID         Type LS ID    Adv Rtr         Seq(Hex) Age  Cksum  Len
0               Inar 0000005b 10.1.1.198      80000001 990  dbad   32
     Options: V6E---R--
     Metric: 1
     Destination Router ID:10.1.1.188
```

**NOTE:** Portions of this display are truncated for brevity. The purpose of this display is to show all possible fields that might display rather than to show complete output.

The fields that display depend upon the LSA type as shown in the following:

**Table 6.3: OSPFv3 detailed database information fields**

| This Field... | Displays... |
|---|---|
| **Router LSA (Type 1) (Rtr) Fields** | |
| Capability Bits | A bit that indicates the capability of the Foundry device. The bit can be set to one of the following:<br>• B – The device is an area border router.<br>• E – The device is an AS boundary router.<br>• V – The device is a virtual link endpoint.<br>• W – The device is a wildcard multicast receiver. |
| Options | A 24-bit field that enables IPv6 OSPF routers to support the optional capabilities. When set, the following bits indicate the following:<br>V6 – The device should be included in IPv6 routing calculations.<br>E – The device floods AS-external-LSAs as described in RFC 2740.<br>MC – The device forwards multicast packets as described in RFC 1586.<br>N – The device handles type 7 LSAs as described in RFC 1584.<br>R – The originator is an active router.<br>DC –The device handles demand circuits. |
| Type | The type of interface. Possible types can be the following:<br>• Point-to-point – A point-to-point connection to another router.<br>• Transit – A connection to a transit network.<br>• Virtual link – A connection to a virtual link. |
| Metric | The cost of using this router interface for outbound traffic. |
| Interface ID | The ID assigned to the router interface. |
| Neighbor Interface ID | The interface ID that the neighboring router has been advertising in hello packets sent on the attached link. |
| Neighbor Router ID | The router ID (IPv4 address) of the neighboring router that advertised the route. (By default, the Foundry router ID is the IPv4 address configured on the lowest numbered loopback interface. If the Foundry device does not have a loopback interface, the default router ID is the lowest numbered IPv4 address configured on the device.) |

**Table 6.3: OSPFv3 detailed database information fields (Continued)**

| This Field... | Displays... |
|---|---|
| **Network LSA (Type 2) (Net) Fields** | |
| Options | A 24-bit field that enables IPv6 OSPF routers to support the optional capabilities. When set, the following bits indicate the following:<br>V6 – The device should be included in IPv6 routing calculations.<br>E – The device floods AS-external-LSAs as described in RFC 2740.<br>MC – The device forwards multicast packets as described in RFC 1586.<br>N – The device handles type 7 LSAs as described in RFC 1584.<br>R – The originator is an active router.<br>DC –The device handles demand circuits. |
| Attached Router | The address of the neighboring router that advertised the route. |
| **Inter-Area Prefix LSA (Type 3) (Inap) Fields** | |
| Metric | The cost of the route. |
| Prefix Options | An 8-bit field describing various capabilities associated with the prefix. |
| Prefix | The IPv6 prefix included in the LSA. |
| **Inter-Area Router LSA (Type 4) (Inar) Fields** | |
| Options | A 24-bit field that enables IPv6 OSPF routers to support the optional capabilities. When set, the following bits indicate the following:<br>V6 – The device should be included in IPv6 routing calculations.<br>E – The device floods AS-external-LSAs as described in RFC 2740.<br>MC – The device forwards multicast packets as described in RFC 1586.<br>N – The device handles type 7 LSAs as described in RFC 1584.<br>R – The originator is an active router.<br>DC –The device handles demand circuits. |
| Metric | The cost of the route. |
| Destination Router ID | The ID of the router described in the LSA. |
| **AS External LSA (Type 5) (Extn) Fields** | |
| Bits | The bit can be set to one of the following:<br>• E – If bit E is set, a Type 2 external metric. If bit E is zero, a Type 1 external metric.<br>• F – A forwarding address is included in the LSA.<br>• T – An external route tag is included in the LSA. |
| Metric | The cost of this route, which depends on bit E. |
| Prefix Options | An 8-bit field describing various capabilities associated with the prefix. |
| Referenced LS Type | If non-zero, an LSA with this LS type is associated with the LSA. |

**Table 6.3: OSPFv3 detailed database information fields (Continued)**

| This Field... | Displays... |
|---|---|
| Prefix | The IPv6 prefix included in the LSA. |
| **Link LSA (Type 8) (Link) Fields** | |
| Router Priority | The router priority of the interface attaching the originating router to the link. |
| Options | The set of options bits that the router would like set in the network LSA that will be originated for the link. |
| Link Local Address | The originating router's link-local interface address on the link. |
| Number of Prefix | The number of IPv6 address prefixes contained in the LSA. |
| Prefix Options | An 8-bit field of capabilities that serve as input to various routing calculations:<br>• NU – The prefix is excluded from IPv6 unicast calculations.<br>• LA – The prefix is an IPv6 interface address of the advertising router.<br>• MC – The prefix is included in IPv6 multicast routing calculations.<br>• P – NSSA area prefixes are readvertised at the NSSA area border. |
| Prefix | The IPv6 prefix included in the LSA. |
| **Intra-Area Prefix LSAs (Type 9) (Iap) Fields** | |
| Number of Prefix | The number of prefixes included in the LSA. |
| Referenced LS Type, Referenced LS ID | Identifies the router-LSA or network-LSA with which the IPv6 address prefixes are associated. |
| Referenced Advertising Router | The address of the neighboring router that advertised the route. |
| Prefix Options | An 8-bit field describing various capabilities associated with the prefix. |
| Metric | The cost of using the advertised prefix. |
| Prefix | The IPv6 prefix included in the LSA. |
| Number of Prefix | The number of prefixes included in the LSA. |

## Displaying OSPFv3 Interface Information

You can display a summary of information for all OSPFv3 interfaces or detailed information about a specified OSPFv3 interface.

To display a summary of OSPFv3 interfaces, enter the following command at any CLI level:

```
BigIron# show ipv6 ospf interface

Interface  OSPF      Status State     Area
ethe 3/1             up
ethe 3/2   enabled   up     DR        0
ethe 3/4   disabled  down
loopback 2 enabled   up     Loopback  0
tunnel 1   disabled  down
tunnel 2   enabled   up     P2P       0
tunnel 6             up
```

***Syntax:*** show ipv6 ospf interface [ethernet <port> | loopback <number> | tunnel <number> | ve <number>]

The **ethernet | loopback | tunnel | ve** parameter specifies the interface for which to display information. If you specify an Ethernet interface, also specify the port number associated with the interface. If you specify a loopback, tunnel, or VE interface, also specify the number associated with the interface.

This display shows the following information:

**Table 6.4: Summary of OSPFv3 interface information**

| This Field... | Displays... |
|---|---|
| Interface | The interface type, and the port number or number of the interface. |
| OSPF | The state of OSPFv3 on the interface. Possible states include the following:<br>• Enabled.<br>• Disabled. |
| Status | The status of the link. Possible status include the following:<br>• Up.<br>• Down. |
| State | The state of the interface. Possible states includes the following:<br>• DR – The interface is functioning as the Designated Router for OSPFv3.<br>• BDR – The interface is functioning as the Backup Designated Router for OSPFv3.<br>• Loopback – The interface is functioning as a loopback interface.<br>• P2P – The interface is functioning as a point-to-point interface.<br>• Passive – The interface is up but it does not take part in forming an adjacency.<br>• Waiting – The interface is trying to determine the identity of the BDR for the network.<br>• None – The interface does not take part in the OSPF interface state machine.<br>• Down – The interface is unusable. No protocol traffic can be sent or received on such a interface.<br>• DR other – The interface is a broadcast or NBMA network on which another router is selected to be the DR. |
| Area | The OSPF area to which the interface belongs. |

For example, to display detailed information about Ethernet interface 2, enter the following command at any level of the CLI:

```
BigIron# show ipv6 ospf interface ethernet 3/2
ethe 3/2 is up, type BROADCAST
  IPv6 Address:
      2002:c0a8:46a::1/64
      2000:4::106/64
  Instance ID 0, Router ID 223.223.223.223
  Area ID 0, Cost 1
  State DR, Transmit Delay 1 sec, Priority 1
  Timer intervals :
    Hello 10, Dead 40, Retransmit 5
  DR:223.223.223.223 BDR:1.1.1.1  Number of I/F scoped LSAs is 2
  DRElection:     5 times, DelayedLSAck:   523 times
  Neighbor Count = 1,   Adjacent Neighbor Count= 1
     Neighbor:
      1.1.1.1 (BDR)
    Statistics of interface ethe 3/2:
      Type       tx   rx tx-byte rx-byte
      Unknown     0    0       0       0
      Hello    3149 3138 1259284 1255352
      DbDesc      7    6     416     288
      LSReq       2    2      80     152
      LSUpdate 1508  530  109508   39036
      LSAck     526 1398   19036   54568
```

This display shows the following information:

**Table 6.5: Detailed OSPFv3 interface information**

| This Field... | Displays... |
|---|---|
| Interface status | The status of the interface. Possible status includes the following:<br>• Up.<br>• Down. |
| Type | The type of OSPFv3 circuit running on the interface. Possible types include the following:<br>• BROADCAST<br>• POINT TO POINT UNKNOWN |
| IPv6 Address | The IPv6 address(es) assigned to the interface. |
| Instance ID | An identifier for an instance of OSPFv3. |
| Router ID | The IPv4 address of the Foundry device. By default, the Foundry router ID is the IPv4 address configured on the lowest numbered loopback interface. If the device does not have a loopback interface, the default router ID is the lowest numbered IPv4 address configured on the device. |
| Area ID | The IPv4 address or numerical value of the area in which the interface belongs. |

**Table 6.5: Detailed OSPFv3 interface information (Continued)**

| This Field... | Displays... |
|---|---|
| Cost | The overhead required to send a packet through the interface. |
| State | The state of the interface. Possible states include the following:<br>• DR – The interface is functioning as the Designated Router for OSPFv3.<br>• BDR – The interface is functioning as the Backup Designated Router for OSPFv3.<br>• Loopback – The interface is functioning as a loopback interface.<br>• P2P – The interface is functioning as a point-to-point interface.<br>• Passive – The interface is up but it does not take part in forming an adjacency.<br>• Waiting – The interface is trying to determine the identity of the BDR for the network.<br>• None – The interface does not take part in the OSPF interface state machine.<br>• Down – The interface is unusable. No protocol traffic can be sent or received on such a interface.<br>• DR other – The interface is a broadcast or NBMA network on which another router is selected to be the DR. |
| Transmit delay | The amount of time, in seconds, it takes to transmit Link State Updates packets on the interface. |
| Priority | The priority used when selecting the DR and the BDR. If the priority is 0, the interface does not participate in the DR and BDR election. |
| Timer intervals | The interval, in seconds, of the hello-interval, dead-interval, and retransmit-interval timers. |
| DR | The router ID (IPv4 address) of the DR. |
| BDR | The router ID (IPv4 address) of the BDR. |
| Number of I/F scoped LSAs | The number of interface LSAs scoped for a specified area, AS, or link. |
| DR Election | The number of times the DR election occurred. |
| Delayed LSA Ack | The number of the times the interface sent a delayed LSA acknowledgement. |
| Neighbor Count | The number of neighbors to which the interface is connected. |
| Adjacent Neighbor Count | The number of neighbors with which the interface has formed an active adjacency. |
| Neighbor | The router ID (IPv4 address) of the neighbor. This field also identifies the neighbor as a DR or BDR, if appropriate. |

**Table 6.5: Detailed OSPFv3 interface information (Continued)**

| This Field... | Displays... |
|---|---|
| Interface statistics | The following statistics are provided for the interface:<br>• Unknown – The number of Unknown packets transmitted and received by the interface. Also, the total number of bytes associated with transmitted and received Unknown packets.<br>• Hello – The number of Hello packets transmitted and received by the interface. Also, the total number of bytes associated with transmitted and received Hello packets.<br>• DbDesc – The number of Database Description packets transmitted and received by the interface. Also, the total number of bytes associated with transmitted and received Database Description packets.<br>• LSReq – The number of link-state requests transmitted and received by the interface. Also, the total number of bytes associated with transmitted and received link-state requests.<br>• LSUpdate – The number of link-state updates transmitted and received by the interface. Also, the total number of bytes associated with transmitted and received link-state requests.<br>• LSAck – The number of link-state acknowledgements transmitted and received by the interface. Also, the total number of bytes associated with transmitted and received link-state acknowledgements. |

## Displaying OSPFv3 Memory Usage

To display information about OSPFv3 memory usage, enter the following command at any level of the CLI:

```
BigIron# show ipv6 ospf memory
Total Static Memory Allocated : 5829 bytes
Total Dynamic Memory Allocated : 0 bytes
Memory Type               Size       Allocated  Max-alloc  Alloc-Fails
MTYPE_OSPF6_TOP           0          0          0          0
MTYPE_OSPF6_LSA_HDR       0          0          0          0
MTYPE_OSPF6_RMAP_COMPILED 0          0          0          0
MTYPE_OSPF6_OTHER         0          0          0          0
MTYPE_THREAD_MASTER       0          0          0          0
MTYPE_OSPF6_AREA          0          0          0          0
MTYPE_OSPF6_AREA_RANGE    0          0          0          0
MTYPE_OSPF6_SUMMARY_ADDRE 0          0          0          0
MTYPE_OSPF6_IF            0          0          0          0
MTYPE_OSPF6_NEIGHBOR      0          0          0          0
MTYPE_OSPF6_ROUTE_NODE    0          0          0          0
MTYPE_OSPF6_ROUTE_INFO    0          0          0          0
MTYPE_OSPF6_PREFIX        0          0          0          0
MTYPE_OSPF6_LSA           0          0          0          0
MTYPE_OSPF6_VERTEX        0          0          0          0
MTYPE_OSPF6_SPFTREE       0          0          0          0
MTYPE_OSPF6_NEXTHOP       0          0          0          0
MTYPE_OSPF6_EXTERNAL_INFO 0          0          0          0
MTYPE_THREAD              0          0          0          0
```

***Syntax:*** show ipv6 ospf memory

This display shows the following information:

**Table 6.6: OSPFv3 memory usage information**

| This Field... | Displays... |
|---|---|
| Total Static Memory Allocated | A summary of the amount of static memory allocated, in bytes, to OSPFv3. |
| Total Dynamic Memory Allocated | A summary of the amount of dynamic memory allocated, in bytes, to OSPFv3. |
| Memory Type | The type of memory used by OSPFv3. (This information is for use by Foundry's technical support in case of a problem.) |
| Size | The size of a memory type. |
| Allocated | The amount of memory currently allocated to a memory type. |
| Max-alloc | The maximum amount of memory that was allocated to a memory type. |
| Alloc-Fails | The number of times an attempt to allocate memory to a memory type failed. |

## Displaying OSPFv3 Neighbor Information

You can display a summary of OSPFv3 neighbor information for the Foundry device or detailed information about a specified neighbor.

To display OSPFv3 neighbor information for the device, enter the following command at any CLI level:

```
BigIron# show ipv6 ospf neighbor
RouterID        Pri State    DR              BDR             Interface[State]
1.1.1.1           1 Full     223.223.223.223 1.1.1.1         ethe 3/2   [DR]
```

***Syntax:*** show ipv6 ospf neighbor [router-id <ipv4-address>]

The **router-id** <ipv4-address> parameter displays only the neighbor entries for the specified router.

This display shows the following information:

**Table 6.7: Summary of OSPFv3 neighbor information**

| Field | Description |
|---|---|
| Router ID | The IPv4 address of the neighbor. By default, the Foundry router ID is the IPv4 address configured on the lowest numbered loopback interface. If the device does not have a loopback interface, the default router ID is the lowest numbered IPv4 address configured on the device. |
| Pri | The OSPFv3 priority of the neighbor. The priority is used during election of the DR and BDR. |

**Table 6.7: Summary of OSPFv3 neighbor information (Continued)**

| Field | Description |
|---|---|
| State | The state between the Foundry device and the neighbor. The state can be one of the following:<br>• Down<br>• Attempt<br>• Init<br>• 2-Way<br>• ExStart<br>• Exchange<br>• Loading<br>• Full |
| DR | The router ID (IPv4 address) of the DR. |
| BDR | The router ID (IPv4 address) of the BDR. |
| Interface [State] | The interface through which the router is connected to the neighbor. The state of the interface can be one of the following:<br>• DR – The interface is functioning as the Designated Router for OSPFv3.<br>• BDR – The interface is functioning as the Backup Designated Router for OSPFv3.<br>• Loopback – The interface is functioning as a loopback interface.<br>• P2P – The interface is functioning as a point-to-point interface.<br>• Passive – The interface is up but it does not take part in forming an adjacency.<br>• Waiting – The interface is trying to determine the identity of the BDR for the network.<br>• None – The interface does not take part in the OSPF interface state machine.<br>• Down – The interface is unusable. No protocol traffic can be sent or received on such a interface.<br>• DR other – The interface is a broadcast or NBMA network on which another router is selected to be the DR. |

For example, to display detailed information about a neighbor with the router ID of 1.1.1.1, enter the following command at any CLI level:

```
BigIron# show ipv6 ospf neighbor router-id 3.3.3.3
RouterID        Pri State    DR              BDR             Interface[State]
3.3.3.3           1 Full     3.3.3.3         1.1.1.1         ve 10    [BDR]
DbDesc bit for this neighbor: --s
Nbr Ifindex of this router: 1
Nbr DRDecision: DR 3.3.3.3, BDR 1.1.1.1
Last received DbDesc: opt:xxx ifmtu:0 bit:--s seqnum:0
Number of LSAs in DbDesc retransmitting: 0
Number of LSAs in SummaryList: 0
Number of LSAs in RequestList: 0
Number of LSAs in RetransList: 0
SeqnumMismatch 0 times, BadLSReq 0 times
OnewayReceived 0 times, InactivityTimer 0 times
DbDescRetrans 0 times, LSReqRetrans 0 times
LSUpdateRetrans 1 times
LSAReceived 12 times, LSUpdateReceived 6 times
```

This display shows the following information:

**Table 6.8: Detailed OSPFv3 neighbor information**

| Field | Description |
|---|---|
| Router ID | For information about this field, see Table 6.7 on page 6-28. |
| Pri | For information about this field, see Table 6.7 on page 6-28. |
| State | For information about this field, see Table 6.7 on page 6-28. |
| DR | For information about this field, see Table 6.7 on page 6-28. |
| BDR | For information about this field, see Table 6.7 on page 6-28. |
| Interface [State] | For information about this field, see Table 6.7 on page 6-28. |
| DbDesc bit... | The Database Description packet, which includes 3 bits of information:<br>• The first bit can be “i” or “-”. “i” indicates the inet bit is set. “-” indicates the inet bit is not set.<br>• The second bit can be “m” or “-”. “m” indicates the more bit is set. “-” indicates the more bit is not set.<br>• The third bit can be “m” or “s”. An “m” indicates the master. An “s” indicates standby. |
| Index | The ID of the LSA from which the neighbor learned of the router. |
| DR Decision | The router ID (IPv4 address) of the neighbor’s elected DR and BDR. |
| Last Received Db Desc | The content of the last database description received from the specified neighbor. |
| Number of LSAs in Db Desc retransmitting | The number of LSAs that need to be retransmitted to the specified neighbor. |
| Number of LSAs in Summary List | The number of LSAs in the neighbor’s summary list. |

**Table 6.8: Detailed OSPFv3 neighbor information (Continued)**

| Field | Description |
|---|---|
| Number of LSAs in Request List | The number of LSAs in the neighbor's request list. |
| Number of LSAs in Retransmit List | The number of LSAs in the neighbor's retransmit list. |
| Seqnum Mismatch | The number of times sequence number mismatches occurred. |
| BadLSReq | The number of times the neighbor received a bad link-state request from the Foundry device. |
| One way received | The number of times a hello packet, which does not mention the router, is received from the neighbor. This omission in the hello packet indicates that the communication with the neighbor is not bidirectional. |
| Inactivity Timer | The number of times that the neighbor's inactivity timer expired. |
| Db Desc Retransmission | The number of times sequence number mismatches occurred. |
| LSReqRetrans | The number of times the neighbor retransmitted link-state requests to the Foundry device. |
| LSUpdateRetrans | The number of times the neighbor retransmitted link-state updates to the Foundry device. |
| LSA Received | The number of times the neighbor received LSAs from the Foundry device. |
| LS Update Received | The number of times the neighbor received link-state updates from the Foundry device. |

## Displaying Routes Redistributed into OSPFv3

You can display all IPv6 routes or a specified IPv6 route that the Foundry device has redistributed into OSPFv3.

To display all IPv6 routes that the device has redistributed into OSPFv3, enter the following command at any level of the CLI:

```
BigIron# show ipv6 ospf redistribute route
Id     Prefix                                    Protocol  Metric Type  Metric
 snIpAsPathAccessListStringRegExpression
1      2002::/16                                 Static    Type-2       1
2      2002:1234::/32                            Static    Type-2       1
```

***Syntax:*** show ipv6 ospf redistribute route [<ipv6-prefix>]

The <ipv6-prefix> parameter specifies an IPv6 network prefix. (You do not need to specify the length of the prefix.)

For example, to display redistribution information for the prefix 2002::, enter the following command at any level of the CLI:

```
BigIron# show ipv6 ospf redistribute route 2002::
Id     Prefix                                    Protocol  Metric Type  Metric
1      2002::/16                                 Static    Type-2       1
```

These displays show the following information:

**Table 6.9: OSPFv3 redistribution information**

| This Field... | Displays... |
|---|---|
| ID | An ID for the redistributed route. |
| Prefix | The IPv6 routes redistributed into OSPFv3. |
| Protocol | The protocol from which the route is redistributed into OSPFv3. Redistributed protocols can be the following:<br>• BGP – BGP4+.<br>• RIP – RIPng.<br>• ISIS – IPv6 IS-IS.<br>• Static – IPv6 static route table.<br>• Connected – A directly connected network. |
| Metric Type | The metric type used for routes redistributed into OSPFv3. The metric type can be the following:<br>• Type-1 – Specifies a small metric (2 bytes).<br>• Type-2 – Specifies a big metric (3 bytes). |
| Metric | The value of the default redistribution metric, which is the OSPF cost of redistributing the route into OSPFv3. |

## Displaying OSPFv3 Route Information

You can display the entire OSPFv3 route table for the Foundry device or only the route entries for a specified destination.

To display the entire OSPFv3 route table for the device, enter the following command at any level of the CLI:

```
  BigIron# show ipv6 ospf routes
Current Route count: 4
    Intra: 4 Inter: 0 External: 0 (Type1 0/Type2 0)
    Equal-cost multi-path: 0
    Destination                    Options   Area            Cost Type2 Cost
    Next Hop Router                Outgoing Interface
*IA 2000:4::/64                    V6E---R-- 0.0.0.0            1 0
    ::                             ethe 3/2
*IA 2002:c0a8:46a::/64             V6E---R-- 0.0.0.0            1 0
    ::                             ethe 3/2
*IA 2999::1/128                    --------- 0.0.0.0            0 0
    ::                             loopback 2
*IA 2999::2/128                    V6E---R-- 0.0.0.0            1 0
    fe80::2e0:52ff:fe91:bb37       ethe 3/2
```

***Syntax:*** show ipv6 ospf routes [<ipv6-prefix>]

The <ipv6-prefix> parameter specifies a destination IPv6 prefix. (You do not need to specify the length of the prefix.) If you use this parameter, only the route entries for this destination are shown.

For example, to display route information for the destination prefix 2000:4::, enter the following command at any level of the CLI:

```
BigIron# show ipv6 ospf routes 2000:4::
Destination                    Options   Area            Cost Type2 Cost
   Next Hop Router                Outgoing Interface
*IA 2000:4::/64                   V6E---R-- 0.0.0.0            1 0
   ::                             ethe 3/2
```

These displays show the following information:

**Table 6.10: OSPFv3 route information**

| This Field... | Displays... |
|---|---|
| Current Route Count (Displays with the entire OSPFv3 route table only) | The number of route entries currently in the OSPFv3 route table. |
| Intra/Inter/External (Type1/Type2) (Displays with the entire OSPFv3 route table only) | The breakdown of the current route entries into the following route types:<br>• Inter – The number of routes that pass into another area.<br>• Intra – The number of routes that are within the local area.<br>• External1 – The number of type 1 external routes.<br>• External2 – The number of type 2 external routes. |
| Equal-cost multi-path (Displays with the entire OSPFv3 route table only) | The number of equal-cost routes to the same destination in the OSPFv3 route table. If load sharing is enabled, the router equally distributes traffic among the routes. |
| Destination | The IPv6 prefixes of destination networks to which the Foundry device can forward IPv6 packets. “*IA” indicates the next router is an intra-area router. |
| Options | A 24-bit field that enables IPv6 OSPF routers to support the optional capabilities. When set, the following bits indicate the following:<br>V6 – The device should be included in IPv6 routing calculations.<br>E – The device floods AS-external-LSAs as described in RFC 2740.<br>MC – The device forwards multicast packets as described in RFC 1586.<br>N – The device handles type 7 LSAs as described in RFC 1584.<br>R – The originator is an active router.<br>DC –The device handles demand circuits. |
| Area | The area whose link state information has led to the routing table entry's collection of paths. |
| Cost | The type 1 cost of this route. |
| Type2 Cost | The type 2 cost of this route. |
| Next-Hop Router | The IPv6 address of the next router a packet must traverse to reach a destination. |

**Table 6.10: OSPFv3 route information (Continued)**

| This Field... | Displays... |
|---|---|
| Outgoing Interface | The router interface through which a packet must traverse to reach the next-hop router. |

## Displaying OSPFv3 SPF Information

You can display the following OSPFv3 SPF information:

- SPF node information for a specified area.
- SPF table for a specified area.
- SPF tree for a specified area.

For example, to display information about SPF nodes in area 0, enter the following command at any level of the CLI:

```
BigIron# show ipv6 ospf spf node area 0
SPF node for Area 0
SPF node 223.223.223.223,  cost: 0,  hops: 0
 nexthops to node:
 parent nodes:
 child nodes: 223.223.223.223:88

SPF node 223.223.223.223:88,  cost: 1,  hops: 1
 nexthops to node:    :: ethe 3/2
 parent nodes: 223.223.223.223
 child nodes: 1.1.1.1:0

SPF node 1.1.1.1:0,  cost: 1,  hops: 2
 nexthops to node:    fe80::2e0:52ff:fe91:bb37 ethe 3/2
 parent nodes: 223.223.223.223:88
 child nodes:
```

***Syntax:*** show ipv6 ospf spf node area [<area-id>]

The **node** keyword displays SPF node information.

The **area** <area-id> parameter specifies a particular area. You can specify the <area-id> in the following formats:

- As an IPv4 address; for example, 192.168.1.1.
- As a numerical value from 0 – 2,147,483,647.

This display shows the following information:

**Table 6.11: OSPFv3 SPF node information**

| This Field... | Displays... |
|---|---|
| SPF node | Each SPF node is identified by its router ID (IPv4 address). If the node is a child node, it is additionally identified by an interface on which the node can be reached appended to the router ID in the format <router-id>:<interface-id>. |
| Cost | The cost of traversing the SPF node to reach the destination. |

**Table 6.11: OSPFv3 SPF node information (Continued)**

| This Field... | Displays... |
|---|---|
| Hops | The number of hops needed to reach the parent SPF node. |
| Next Hops to Node | The IPv6 address of the next hop-router and/or the router interface through which to access the next-hop router. |
| Parent Nodes | The SPF node's parent nodes. A ***parent node*** is an SPF node at the highest level of the SPF tree, which is identified by its router ID. |
| Child Nodes | The SPF node's child nodes. A ***child node*** is an SPF node at a lower level of the SPF tree, which is identified by its router ID and interface on which the node can be reached. |

For example, to display the SPF table for area 0, enter the following command at any level of the CLI:

```
BigIron# show ipv6 ospf spf table area 0
  SPF table for Area 0
  Destination         Bits Options  Cost  Nexthop                    Interface
R 1.1.1.1             ---- V6E---R-    1  fe80::2e0:52ff:fe91:bb37  ethe 3/2
N 223.223.223.223[88] ---- V6E---R-    1  ::                         ethe 3/2
```

***Syntax:*** show ipv6 ospf spf table area <area-id>

The **table** parameter displays the SPF table.

The **area** <area-id> parameter specifies a particular area. You can specify the <area-id> in the following formats:

- As an IPv4 address, for example, 192.168.1.1.
- As a numerical value from 0 – 2,147,483,647.

This display shows the following information:

**Table 6.12: OSPFv3 SPF Table**

| This Field... | Displays... |
|---|---|
| Destination | The destination of a route, which is identified by the following:<br>• "R", which indicates the destination is a router. "N", which indicates the destination is a network.<br>• An SPF node's router ID (IPv4 address). If the node is a child node, it is additionally identified by an interface on which the node can be reached appended to the router ID in the format <router-id>:<interface-id>. |
| Bits | A bit that indicates the capability of the Foundry device . The bit can be set to one of the following:<br>• B – The device is an area border router.<br>• E – The device is an AS boundary router.<br>• V – The device is a virtual link endpoint.<br>• W – The device is a wildcard multicast receiver. |

**Table 6.12: OSPFv3 SPF Table (Continued)**

| This Field... | Displays... |
|---|---|
| Options | A 24-bit field that enables IPv6 OSPF routers to support the optional capabilities. When set, the following bits indicate the following:<br>V6 – The router should be included in IPv6 routing calculations.<br>E – The router floods AS-external-LSAs as described in RFC 2740.<br>MC – The router forwards multicast packets as described in RFC 1586.<br>N – The router handles type 7 LSAs as described in RFC 1584.<br>R – The originator is an active router.<br>DC –The router handles demand circuits. |
| Cost | The cost of traversing the SPF node to reach the destination. |
| Next hop | The IPv6 address of the next hop-router. |
| Interface | The router interface through which to access the next-hop router. |

For example, to display the SPF tree for area 0, enter the following command at any level of the CLI:

```
BigIron# show ipv6 ospf spf tree area 0
   SPF tree for Area 0
 +- 223.223.223.223 cost 0
     +- 223.223.223.223:88 cost 1
         +- 1.1.1.1:0 cost 1
```

***Syntax:*** show ipv6 ospf spf tree area <area-id>

The **tree** keyword displays the SPF table.

The **area** <area-id> parameter specifies a particular area. You can specify the <area-id> in the following formats:

- As an IPv4 address; for example, 192.168.1.1.
- As a numerical value from 0 – 2,147,483,647.

In this sample output, consider the SPF node with the router ID 223.223.223.223 to be the top (root) of the tree and the local router. Consider all other layers of the tree (223.223.223.223:88 and 1.1.1.1:0) to be destinations in the network. Therefore, traffic destined from router 223.223.223.223 to router 1.1.1.1:0 must first traverse router 223.223.223.223:88.

## Displaying IPv6 OSPF Virtual Link Information

To display OSPFv3 virtual link information for the Foundry device, enter the following command at any level of the CLI:

```
BigIron# show ipv6 ospf virtual-link
Index Transit Area ID  Router ID       Interface Address            State
1     1                1.1.1.1         3003::2                      P2P
```

***Syntax:*** show ipv6 ospf virtual-link

This display shows the following information:

**Table 6.13: OSPFv3 virtual link information**

| This Field... | Displays... |
|---|---|
| Index | An index number associated with the virtual link. |
| Transit Area ID | The ID of the shared area of two ABRs that serves as a connection point between the two routers. |
| Router ID | IPv4 address of the router at the other end of the virtual link (virtual neighbor). |
| Interface Address | The local address used to communicate with the virtual neighbor. |
| State | The state of the virtual link. Possible states include the following:<br>• P2P – The link is functioning as a point-to-point interface.<br>• DOWN – The link is down. |

## Displaying OSPFv3 Virtual Neighbor Information

To display OSPFv3 virtual neighbor information for the Foundry device, enter the following command at any level of the CLI:

```
BigIron# show ipv6 ospf virtual-neighbor
Index Router ID        Address                       State     Interface
1     1.1.1.1          3002::1                       Full      ethe 2/3
```

***Syntax:*** show ipv6 ospf virtual-neighbor

This display shows the following information:

**Table 6.14: OSPFv3 virtual neighbor information**

| This Field... | Displays... |
|---|---|
| Index | An index number associated with the virtual neighbor. |
| Router ID | IPv4 address of the virtual neighbor. |
| Address | The IPv6 address to be used for communication with the virtual neighbor. |

**Table 6.14: OSPFv3 virtual neighbor information (Continued)**

| This Field... | Displays... |
|---|---|
| State | The state between the Foundry device and the virtual neighbor. The state can be one of the following:<br>• Down<br>• Attempt<br>• Init<br>• 2-Way<br>• ExStart<br>• Exchange<br>• Loading<br>• Full |
| Interface | The IPv6 address of the virtual neighbor. |

# Chapter 7
# Configuring IPv6 IS-IS

**NOTE:** IPv6 ISIS is supported on the NetIron 40G beginning with Terathon software release 02.2.01.

The Intermediate System to Intermediate System (IS-IS) protocol is a link-state Interior Gateway Protocol (IGP) that is based on the International Standard for Organization/International Electrotechnical Commission (ISO/IEC) Open Systems Internet Networking model (OSI). In IS-IS, an intermediate system (router) is designated as either a Level 1 or Level 2 router. A Level 1 router routes traffic only within the area in which the router resides. A Level 2 router routes traffic between areas within a routing domain.

**NOTE:** This section provides information about configuring IPv6 IS-IS only. For information about configuring IPv4 IS-IS, see the "Configuring IS-IS" chapter in the *Foundry NetIron Service Provider Configuration and Management Guide*

The Foundry implementation of IS-IS is based on the following specifications and draft specifications:

- ISO/IEC 10589 – "Information Technology – Telecommunication and information exchange between systems – Intermediate system to Intermediate system intra-domain routing information exchange protocol for use in conjunction with the protocol for providing the connection less-mode Network Service (ISO 8473)", 1992
- ISO/IEC 8473 – "Information processing systems – Data Communications – Protocols for providing the connectionless-mode network service", 1988
- ISO/IEC 9542 – "Information Technology – Telecommunication and information exchange between systems – End system to Intermediate system intra-domain routing information exchange protocol for use in conjunction with the protocol for providing the connection less-mode Network Service (ISO 8473)", 1988
- RFC 1195 – "Use of OSI IS-IS for Routing in TCP/IP and Dual Environments", 1990.
- RFC 1377 – "The PPP OSI Network Layer Control Protocol (OSINLCP)", 1992.
- RFC 2763 – "Dynamic Host Name Exchange Mechanism for IS-IS", 2000.
- RFC 2966 – "Domain-wide Prefix Distribution with Two-Level IS-IS", 2000
- Portions of the Internet Draft "IS-IS extensions for Traffic Engineering" (dated 2000). that describe the Extended IP reachability TLV (TLV type 135) and the extended Intermediate System (IS) reachability TLV (TLV type 22). These portions provide support for the wide metric version of IS-IS. No other portion is supported on Foundry's implementation of IS-IS.

**NOTE:** The Layer 3 Switch does not support routing of Connectionless-Mode Network Protocol (CLNP) packets. The Layer 3 Switch uses IS-IS for TCP/IP only.

## Relationship to IP Route Table

The IS-IS protocol has the same relationship to the Layer 3 Switch's IP route table that OSPF has to the table. The protocol sends the best IS-IS path to a given destination to the CPU for comparison to the best paths from other protocols to the same destination. The CPU selects the path with the lowest administrative distance and places that path in the IP route table.

- If the path provided by IS-IS has the lowest administrative distance, then the CPU places that IS-IS path in the IP route table.
- If a path to the same destination supplied by another protocol has a lower administrative distance, the CPU installs the other protocol's path in the IP route table instead.

The ***administrative distance*** is a protocol-independent value from 1 – 255. Each path sent to the CPU, regardless of the source of the path (IS-IS, OSPF, static IP route, and so on) has an administrative distance.

Each route source has a default administrative distance. The default administrative distance for IS-IS is 115.

You can change the administrative distance for IS-IS and other routes sources.

## Intermediate Systems and End Systems

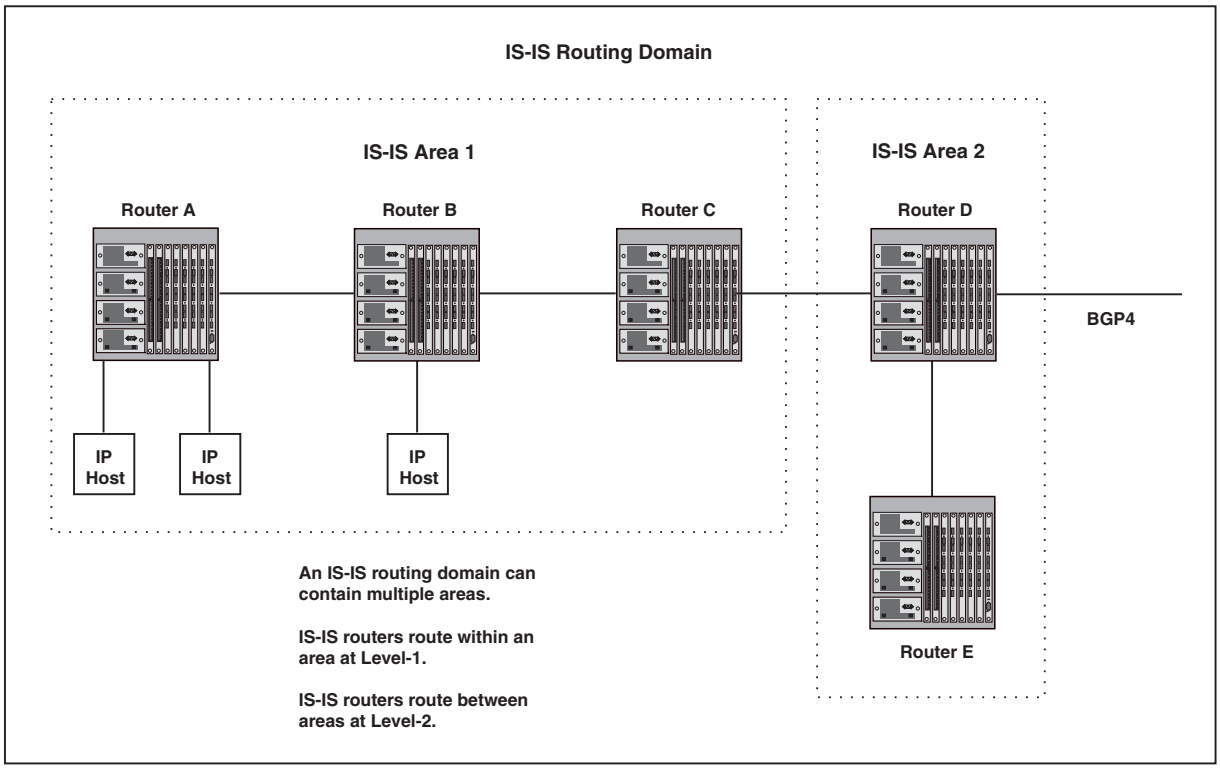
IS-IS uses the following categories to describe devices within an IS-IS routing domain (similar to an OSPF Autonomous System):

- ***Intermediate System (IS)*** – A device capable of forwarding packets from one device to another within the domain. In Internet Protocol (IP) terminology, an IS is a router. (Foundry routers are called "Layer 3 Switches".)
- ***End System (ES)*** – A device capable of generating or receiving packets within the domain. In IP terminology, an ES is an end node or IP host.

When you configure IS-IS on a Foundry Layer 3 Switch, the device is an IS.

Figure 7.1 shows an example of an IS-IS network.

**Figure 7.1An IS-IS network contains Intermediate Systems (ISs) and host systems**




**NOTE:** Since the Foundry implementation of IS-IS does not route OSI traffic but instead routes IP traffic, IP hosts are shown instead of ESs.

The other basic IS-IS concepts illustrated in this figure are explained in the following sections.

## Domain and Areas

IS-IS is an IGP, and thus applies only to routes within a single routing domain. However, you can configure multiple areas within a domain. A Foundry Layer 3 Switch can be a member of one area for each Network Entity Title (NET) you configure on the Layer 3 Switch. The NET contains the area ID for the area the NET is in.

In Figure 7.1, Routers A, B, and C are in area 1. Routers D and E are in area 2. All the routers are in the same domain.

## Level-1 Routing and Level-2 Routing

You can configure an IS-IS router such as a Foundry Layer 3 Switch to perform one or both of the following levels of IS-IS routing[1]:

- Level-1 – A Level-1 router routes traffic only within the area the router is in. To forward traffic to another area, the Level-1 router sends the traffic to its nearest Level-2 router.
- Level-2 – A Level-2 router routes traffic between areas within a domain.

In Figure 7.1 on page 7-3, Routers A and B are Level-1 ISs only. Routers C and D are Level-1 ISs and Level-2 ISs. Router E is a Level-1 ISs only.

1.The ISO/IEC specifications use the spelling “routeing”, but this document uses the spelling “routing” to remain consistent with other Foundry documentation.

## Neighbors and Adjacencies

A Layer 3 Switch configured for IS-IS forms an ***adjacency*** with each of the IS-IS devices to which it is directly connected. An adjacency is a two-way direct link (a link without router hops) over which the two devices can exchange IS-IS routes and other protocol-related information. The link is sometimes called a “circuit”. The devices with which the Layer 3 Switch forms adjacencies are its IS-IS ***neighbors***, which are other ISs.

A circuit can be a broadcast circuit or a point-to-point circuit. Foundry IS-IS interfaces are configured by default for broadcast circuits, but you can change the circuit type on an interface to point-to-point. Each end of an IS-IS adjacency must use the same circuit type.

In Figure 7.1 on page 7-3, Router A has an IS-IS adjacency with Router B. Likewise, Router B has an IS-IS adjacency with Router A and Router C.

## Designated IS

A ***Designated IS*** is an IS-IS router that is responsible for gathering and distributing link state information to other Level-1 or Level-2 ISs within the same broadcast network (LAN). The Level-1 and Level-2 Designated ISs within a broadcast network are independent, although the same Layer 3 Switch can be a Level-1 Designated IS and a Level-2 Designated IS at the same time.

The Designated IS is elected based on the priority of each IS in the broadcast network. When an IS becomes operational, it sends a Level-1 or Level-2 Hello PDU to advertise itself to other ISs. If the IS is configured to be both a Level-1 and a Level-2 IS, the IS sends a separate advertisement for each level.

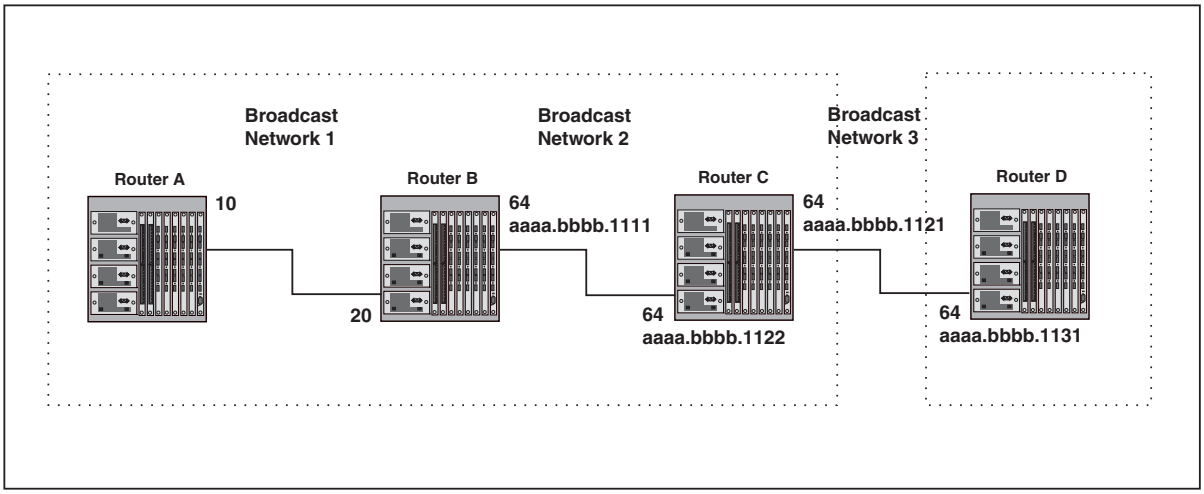
- The Level-1 IS that has the highest priority becomes the Level-1 Designated IS for the broadcast network.
- The Level-2 IS that has the highest priority becomes the Level-2 Designated IS for the broadcast network.

If the Designated IS becomes unavailable (for example, is rebooted), the IS with the next highest priority becomes the new IS. If two or more ISs have the highest priority, the IS with the highest MAC address becomes the Designated IS.

The priority is an interface parameter. Each interface that is enabled for IS-IS can have a different priority.

Figure 7.2 shows an example of the results of Designated IS elections. For simplicity, this example shows four of the five routers in Figure 7.1 on page 7-3, withe same domain and areas.

**Figure 7.2Each broadcast network has a Level-1 Designated IS and a Level-2 Designated IS**




Designated IS election has the following results in this network topology:

- Router B is the Level-1 Designated IS for broadcast network 1
- Router C is the Level-1 Designated IS for broadcast network 2
- Router D is the Level-2 Designated IS for broadcast network 3

In this example, the IS-IS priorities for the IS-IS interfaces in broadcast network 1 have been changed by an administrator. The priorities for the interfaces in the other broadcast networks are still set to the default (64). When there is a tie, IS-IS selects the interface with the highest MAC address.

### Broadcast Pseudonode

In a broadcast network, the Designated IS maintains and distributes link state information to other ISs by maintaining a ***pseudonode***. A pseudonode is a logical host representing all the Level-1 or Level-2 links among the ISs in a broadcast network. Level-1 and Level-2 have separate pseudonodes, although the same device can be the pseudonode for Level-1 and Level-2.

### Route Calculation and Selection

The Designated IS uses a ***Shortest Path First (SPF)*** algorithm to calculate paths to destination ISs and ESs. The SPF algorithm uses Link State PDUs (LSPDUs) received from other ISs as input, and creates the paths as output.

After calculating the paths, the Designated IS then selects the best paths and places them in the IS-IS route table. The Designated IS uses the following process to select the best paths:

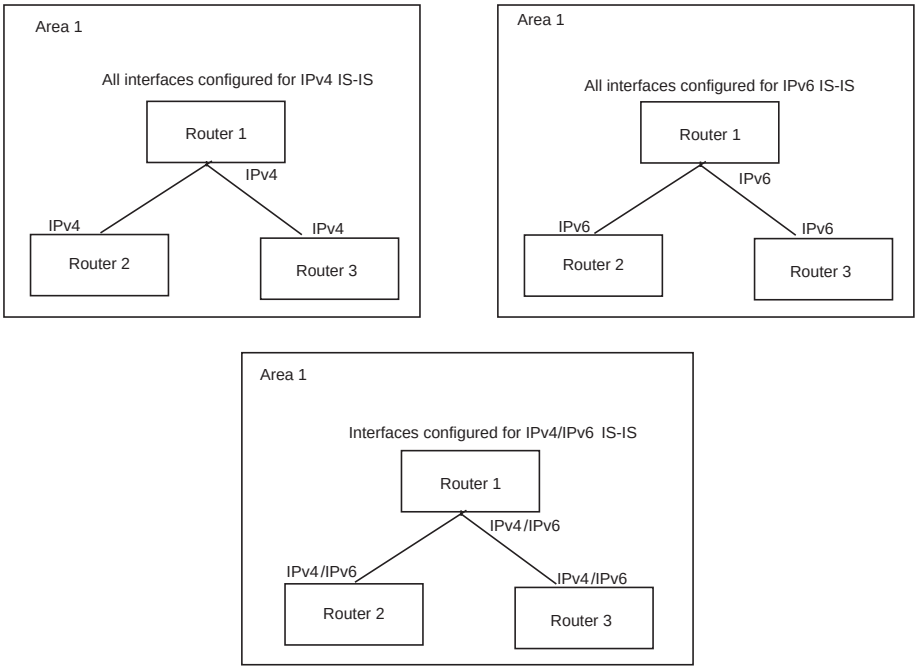
1. Prefer the Level-1 path over the Level-2 path.
2. If there is no Level-1 path, prefer the internal Level-2 path over the external Level-2 path.
3. If there is still more than one path, prefer the path with the lowest metric.
4. If there is more than one path with the lowest metric, load share among the paths.

After selecting the best path to a destination, the software places the path in the IS-IS route table.

## IPv6 IS-IS Single-Topology Mode

IPv6 IS-IS supports single-topology mode, which means that you can run IPv6 IS-IS concurrently with other network protocols such as IPv4 IS-IS throughout a topology. However, when implementing a single topology, all routers in an area (Level 1 routing) or domain (Level 2 routing) must be configured with the same set of network protocols on all its interfaces, even on loopback interfaces. You can configure IPv4 IS-IS only, IPv6 IS-IS only, or both IPv4 IS-IS and IPv6 IS-IS ((Figure 7.3). For example, to successfully implement both IPv4 and IPv6 IS-IS in an area, you must configure both IPv4 and IPv6 IS-IS on all router interfaces in the area.

**Figure 7.3**




A single shortest path first (SPF) per level computes the IPv4 and IPv6 routes. The use of a single SPF indicates that both IPv4 and IPv6 IS-IS routing protocols must share a common network topology

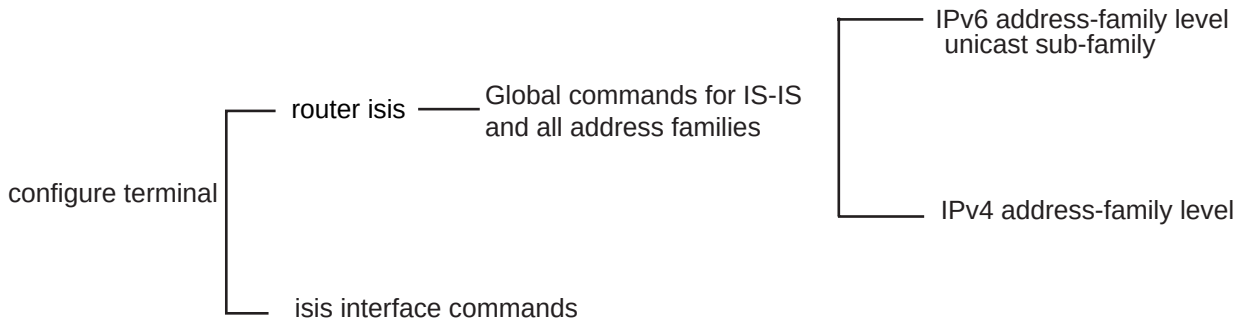
Foundry's implementation of IPv4 IS-IS supports type, length, and value (TLV) parameters to advertise reachability to IPv4 networks. The TLVs specify the types of data, the maximum length of the data, and the valid values for the data. IPv6 IS-IS advertises its information using new TLV parameters. The new TLV parameters for IPv6 support an extended default metric value.

When IPv6 IS-IS is enabled in a single topology, you must set the value of the **metric** command to wide at the interface level. Wide is the default for IPv6. If both IPv4 and IPv6 are configured on an interface, metric must be set to wide on all interfaces. Narrow is the default for IPv4.

# IS-IS CLI Levels

The CLI includes various levels of commands for IS-IS. Figure 7.4 diagrams these levels.

**Figure 7.4IPv6 IS-IS CLI Levels**




The IPv6 IS-IS CLI levels are as follows:

- A global level for the configuration of the IS-IS protocol. At this level, all IS-IS configurations at this level apply to IPv4 and IPv6. You enter this layer using the **router isis** command.
    - Under the global level, you specify an address family. Address families separate the IS-IS configuration IPv6 and IPv4. You enter configurations that are for a specific You enter this level by entering the **address-family** command at the router isis level.
    - Under the address family level, you select a sub-address family, which is the type of routes for the configuration. For IS-IS, you specify **unicast**.
- An interface level

## Global Configuration Level

You enter the global configuration level of ISIS by entering the following command:

```
BigIron MG8(config)#router isis
BigIron MG8(config-isis-router)#
```

***Syntax:*** router isis

The `(config-isis-router)#` prompt indicates that you are at the global level for IS-IS. Configurations you enter at this level apply to both IS-IS IPv4 and IS-IS IPv6.

## Address Family Configuration Level

Foundry's implementation of IPv6 IS-IS includes a new configuration level: address family. You enter IS-IS definitions for IPv6 IS-IS under this level. Address-family allows you to create configurations for IPv6 IS-IS unicast routes that are separate and distinct from configurations for IPv4 IS-IS unicast routes.

Under the address family level, Foundry currently supports the unicast address family configuration level only. The Foundry device enters the IPv6 IS-IS unicast address family configuration level when you enter the following command while at the global IS-IS configuration level:

```
BigIron MG8(config-isis-router)# address-family ipv6 unicast
BigIron MG8(config-isis-router-ipv6u)#
```

***Syntax:*** address-family ipv6 unicast

The `(config-isis-router-ipv6u)#` prompt indicates that you are at the IPv6 IS-IS unicast address family configuration level. While at this level, you can access several commands that allow you to configure IPv6 IS-IS unicast routes.

---

**NOTE:** Each address family configuration level allows you to access commands that apply to that particular address family only. To enable a feature in a particular address family, you must specify any associated commands for that feature in that particular address family. You cannot expect the feature, which you may have configured in the IPv4 IS-IS unicast address family, to work in the IPv6 IS-IS unicast address family unless it is explicitly configured in the IPv6 IS-IS unicast address family.

---

To exit from the IPv6 IS-IS unicast address family configuration level, enter the following command:

```
BigIron MG8(config-isis-router-ipv6u)# exit-address-family
BigIron MG8(config-isis-router)#
```

Entering this command returns you to the global IS-IS configuration level.

## Interface Level

Some IS-IS definitions are entered at the interface level. To change to the interface level for IS-IS configuration, enter the following command.

```
BigIron MG8(config)# interface ethernet 2/3
BigIron MG8(config-if-e1000-2/3)#ipv6 router isis
```

***Syntax:*** ipv6 router isis

# Configuring IPv6 IS-IS

## Enabling IS-IS Globally

To configure IPv6 IS-IS, do the following

1. You must enable the forwarding of IPv6 traffic on the Foundry device using the **ipv6 unicast-routing** command. Enter a command such as the following:

   ```
   BigIron MG8#configure terminal
   BigIron MG8(config)# ipv6 unicast-routing
   ```

   ***Syntax:*** [no] ipv6 unicast-routing

2. Globally enable IS-IS by entering the following command:

   ```
   BigIron MG8(config)# router isis
   ISIS: Please configure NET!
   ```

   Once you enter **router isis**, the device enters the IS-IS router configuration level.

   ***Syntax:*** [no] router isis

   To disable IS-IS, use the **no** form of this command.

3. If you have not already configured a NET for IS-IS, enter commands such as the following:

   ```
   BigIron MG8(config-isis-router)# net 49.2211.aaaa.bbbb.cccc.00
   BigIron MG8(config-isis-router)#
   ```

   The commands in the example above configure a NET that has the area ID 49.2211, the system ID aaaa.bbbb.cccc (the device's base MAC address), and SEL value 00.

   ***Syntax:*** [no] net <area-id>.<system-id>.<sel>

   The <area-id> parameter specifies the area and has the format xx or xx.xxxx. For example, 49 and 49.2211 are valid area IDs.

   The <system-id> parameter specifies the router's unique IS-IS router ID and has the format xxxx.xxxx.xxxx. You can specify any value for the system ID. A common practice is to use the device's base MAC address as the system ID. The base MAC address is also the MAC address of port 1. To determine the base MAC address, enter the following command at any level of the CLI: **show interfaces brief**. The base MAC address is listed in the first row of information, in the MAC column.

   You must use the same system ID in all the NETs on the Foundry device.

   ---

   **NOTE:** The parameter descriptions above are the recommended values for the NET. However, the CLI accepts any value that fits within the following lengths and formats:

   xx.xxxx.xxxx.xxxx.00 – minimum length of NET

   xx.xxxx.xxxx.xxxx.xxxx.xxxx.xxxx.xxxx.xxxx.xxxx.00 – maximum length of NET

   ---

   The <sel> parameter specifies the NSAP Selector (SEL). This value must always be 00 (two zeros). The value 00 indicates that this address is an NET.

   To delete a NET, use the **no** form of this command.

4. Configure an IPv6 IS-IS single topology. See "Configuring IPv6 IS-IS Single Topology" on page 7-9.

5. Configure ISIS parameters. See the sections "Globally Configuring IS-IS on a Device" on page 7-9, "Configuring IPv6 Address Family Route Parameters" on page 7-14, and "Configuring ISIS Properties on an Interface" on page 7-20.

### Enabling IS-IS and Assigning an IPv6 Address to an Interface

To configure IPv6 IS-IS on the desired router interfaces, enter commands such as the following:

```
BigIron MG8(config)# interface ethernet 3/1
BigIron MG8(config-if-e100-3/1)# ipv6 address 2001:200:12D:1300::/64 eui-64
BigIron MG8(config-if-e100-3/1)# ipv6 router isis
```

The commands in this example assign the global IPv6 prefix 2001:200:12d:1300::/64 to Ethernet interface 3/1 and enable IPv6 IS-IS on the interface.

***Syntax:*** ipv6 address <ipv6-prefix>/<prefix-length> [eui-64]

You must specify the <ipv6-prefix> parameter in hexadecimal using 16-bit values between colons as documented in RFC 2373.

You must specify the <prefix-length> parameter as a decimal value. A slash mark (/) must follow the <ipv6-prefix> parameter and precede the <prefix-length> parameter.

The **eui-64** keyword configures the global or site-local address with an EUI-64 interface ID in the low-order 64 bits. The interface ID is automatically constructed in IEEE EUI-64 format using the interface's MAC address.

***Syntax:*** [no] ipv6 router isis

To disable IPv6 IS-IS on an interface, use the **no** form of this command.

The following configuration tasks are optional:

- Configure IPv6 route parameters.
- Redistribute routes from other route sources into IPv6 IS-IS.
- Perform IPv6 IS-IS adjacency checks.
- Disable partial SPF calculations

## Configuring IPv6 IS-IS Single Topology

If your IS-IS single topology will support both IPv6 and IPv4, you can configure both IPv6 and IPv4 on an IS-IS interface for Level 1, Level 2, or both Level 1 and Level 2. However, if you configure both IPv6 and IPv4 on an IS-IS interface, they must be configured to run on the same level. For example, you can configure IPv6 to run on Level 1 on an interface and IPv4 to also run on Level 1 on the same interface. However, you cannot configure IPv6 to run on Level 1 on an interface and IPv4 to run to Level 2 on the same interface.

To configure an IPv6 IS-IS single topology, you must do the following:

1. Globally enable IS-IS and configure at least one Network Entity Title (NET). The NET is the Foundry device's network interface with IS-IS. You can configure up to three NETs on a device.
2. Configure the desired router interfaces with an IPv6 address and enable IPv6 IS-IS on the router interfaces.
3. Configure ISIS parameters. See the sections "Globally Configuring IS-IS on a Device" on page 7-9, "Configuring IPv6 Address Family Route Parameters" on page 7-14, and "Configuring ISIS Properties on an Interface" on page 7-20.

## Globally Configuring IS-IS on a Device

This section describes how to change the global IS-IS parameters. These parameter settings apply to both IS-IS IPv4 and IS-IS IPv6.

### Setting the Overload Bit

If an IS's resources are overloaded and are preventing the IS from properly performing IS-IS routing, the IS can inform other ISs of this condition by setting the overload bit in LSPDUs sent to other ISs from 0 (off) to 1 (on).

When an IS is overloaded, other ISs will not use the overloaded IS to forward traffic. An IS can be in the overload state for Level-1, Level-2, or both.

- If an IS is in the overload state for Level-1, other Level-1 ISs stop using the overloaded IS to forward Level-1 traffic. However, the IS can still forward Level-2 traffic, if applicable.
- If an IS is in the overload state for Level-2, other Level-2 ISs stop using the overloaded IS to forward Level-2 traffic. However, the IS can still forward Level-1 traffic, if applicable.
- If an IS is in the overload state for both levels, the IS cannot forward traffic at either level.

By default, the Layer 3 Switch automatically sets the overload bit to 1 (on) in its LSPDUs to other ISs if an overload condition occurs.

You can set the overload bit on to administratively shut down IS-IS without disabling the protocol. Setting the overload bit on is useful when you want to make configuration changes without removing the Layer 3 Switch from the network.

In addition, you can configure the Layer 3 Switch to set the overload bit on for a specific number of seconds during startup, to allow IS-IS to become fully active before the device begins IS-IS routing. By default, there is no delay (0 seconds).

To immediately set the overload bit on, enter the following command:

```
BigIron MG8(config-isis-router)# set-overload-bit
```

This command administratively shuts down IS-IS by configuring the Layer 3 Switch to immediately set the overload bit to 1 (on) in all LSPs sent to other ISs.

To configure the Layer 3 Switch to temporarily set the overload bit on after a software reload, enter a command such as the following:

```
BigIron MG8(config-isis-router)# set-overload-bit on-startup 5
```

This command configures the Layer 3 Switch to set the overload bit on in all IS-IS LSPs sent to other ISs during the first five seconds following a successful software reload. After the five seconds expire, the Layer 3 Switch stops setting the overload bit on, and instead starts setting the overload bit off.

***Syntax:*** [no] set-overload-bit [on-startup <secs>]

The **on-startup** <secs> parameter specifies the number of seconds following a reload to set the overload bit on. You can specify 0 or a number from 5 – 86400 (24 hours). The default is 0, which means the Layer 3 Switch starts performing IS-IS routing immediately following a successful software reload.

## Configuring Authentication

By default, the Layer 3 Switch does not authenticate packets sent to or received from ESs or other ISs. You can configure the following types of passwords for IS-IS globally.

**Table 7.1: IS-IS Passwords**

| Password Type | Scope | Where Used | Default |
|---|---|---|---|
| Domain | Level-2 | Level-2 LSPDU | None configured |
| Area | Level-1 | Level-1 LSPDU | None configured |
| Interface | Level-1 and Level-2 | Hello PDU | None configured |

If you configure a password, the Layer 3 Switch checks for the password in IS-IS packets received by the device and includes the password in packets sent by the device. For example, the Layer 3 Switch checks all Level-2 LSPDUs received by the device for the domain password you configure, and includes the password in all Level-2 PDUs sent by the device.

### Configuring a Domain Password

To configure an IS-IS domain password, enter a command such as the following:

```
BigIron MG8(config-isis-router)# domain-password domain-1
```

This command configures the Foundry device to use the password “domain-1” to authenticate Level-2 LSPDUs.

***Syntax:*** [no] domain-password <string>

The <string> parameter specifies the password. You can enter an alphanumeric string up to 80 characters long. The password can contain blank spaces. If you use a blank space in the password, you must use quotation marks (“ “) around the entire password; for example, **domain-password “domain 1”**.

### Configuring an Area Password

To configure an IS-IS area password, enter a command such as the following:

```
BigIron MG8(config-isis-router)# area-password area-51
```

This command configures the Foundry device to use the password “area-51” to authenticate Level-1 LSPDUs.

***Syntax:*** [no] area-password <string>

The <string> parameter specifies the password. You can enter an alphanumeric string up to 80 characters long. The password can contain blank spaces. If you use a blank space in the password, you must use quotation marks (“ “) around the entire password; for example, **area-password “area 51”**.

## Changing the IS-IS Level Globally

By default, a Foundry Layer 3 Switch can operate as both a Level-1 and IS-IS Level-2 router. To globally change the type of IS-IS packets supported on the device from Level-1 and Level-2 to Level-1 only, enter the following command:

```
BigIron MG8(config-isis-router)# is-type level-1-only
```

***Syntax:*** [no] is-type level-1-only | level-1-2 | level-2-only

The **level-1-only | level-1-2 | level-2-only** parameter specifies the IS-IS type. If you want to re-enable support for both IS-IS types, re-enter the command you entered to change the IS-IS type, and use “no” in front of the command.

To change the IS-IS on an interface, see “Changing the IS-IS Level on an Interface” on page 7-21.

## Disabling or Re-enabling Display of Layer 3 Switch Hostname

Foundry’s implementation of IS-IS supports RFC 2763, which describes a mechanism for mapping IS-IS system IDs to the hostnames of the devices with those IDs. For example, if you set the hostname on the Layer 3 Switch to “IS-IS Router 1”, the mapping feature uses this name instead of the Layer 3 Switch’s IS-IS system ID in the output of the following commands:

- **show isis database**
- **show isis interface**
- **show isis neighbor**

The Layer 3 Switch’s hostname is displayed in each CLI command prompt, for example:

```
IS-IS Router 1(config-isis-router)#
```

The name mapping feature is enabled by default. If you want to disable name mapping, enter the following command:

```
IS-IS Router 1(config-isis-router)# no hostname
```

***Syntax:*** [no] hostname

To display the name mappings, enter the **show isis hostname** command. See the *Foundry NetIron Service Provider Configuration and Management Guide* for information on IS-IS IPv4 and global IS-IS show commands.

## Changing the Sequence Numbers PDU Interval

A ***Complete Sequence Numbers PDU (CSNP)*** is a complete list of the LSPs in the Designated IS' link state database. The CSNP contains a list of all the LSPs in the database, as well as other information that helps IS neighbors determine whether their LSP databases are in sync with one another. The Designated IS sends CSNPs to the broadcast interface. Level-1 and Level-2 each have their own Designated IS.

A ***Partial Sequence Numbers PDU (PSNP)*** is a partial list of LSPs. ISs other than the Designated IS (that is, the non-Designated ISs) send PSNPs to the broadcast interface.

The CSNP interval specifies how often the Designated IS sends a CSNP to the broadcast interface. Likewise, the PSNP interval specifies how often other ISs (non-Designated ISs) send a PSNP to the broadcast interface. (The PSNP interval also applies to ISs on a point-to-point network.)

The interval you can configure on the Layer 3 Switch applies to both Level-1 and Level-2 CSNPs and PSNPs. The default interval is 10 seconds. You can set the interval to a value from 0 – 65535 seconds.

To change the interval, enter a command such as the following:

```
BigIron MG8(config-isis-router)# csnp-interval 15
```

***Syntax:*** [no] csnp-interval <secs>

The <secs> parameter specifies the interval and can be from 0 – 65535 seconds. The default is 10 seconds.

**NOTE:** Although the command name is **csnp-interval**, the interval also applies to PSNPs.

## Changing the Maximum LSP Lifetime

The maximum LSP lifetime is the maximum number of seconds an unrefreshed LSP can remain in the Layer 3 Switch's LSP database. The maximum LSP lifetime can be from 1 – 65535 seconds. The default is 1200 seconds (20 minutes).

To change the maximum LSP lifetime to 2400 seconds, enter a command such as the following:

```
BigIron MG8(config-isis-router)# max-lsp-lifetime 2400
```

***Syntax:*** [no] max-lsp-lifetime <secs>

The <secs> parameter specifies the maximum LSP lifetime and can be from 1 – 65535 seconds. The default is 1200 seconds (20 minutes).

## Changing the LSP Interval and Retransmit Interval

You LSP interval is the rate of transmission, in seconds of the LSPs. The retransmit interval is the time the device waits before it retransmits LSPs. To define an LSP interval, enter a command such as the following:

```
BigIron MG8(config-isis-router)# lsp-interval 45
```

***Syntax:*** [no] lsp interval <seconds>

Enter 1 – 4294967295 seconds for the LSP interval.

To define an interval for retransmission of LSPs enter a command such as the following:

```
BigIron MG8(config-isis-router)#retransmit-interval 3
```

***Syntax:*** [no] retransmit-interval

Enter 0 – 65535 seconds for the retransmission interval.

## Changing the LSP Refresh Interval

The LSP refresh interval is the maximum number of seconds the Layer 3 Switch waits between sending updated LSPs to its IS-IS neighbors. The interval can be from 1 – 65535 seconds. The default is 900 seconds.

To change the LSP refresh interval to 20000 seconds, enter a command such as the following:

```
BigIron MG8(config-isis-router)# lsp-refresh-interval 20000
```

***Syntax:*** [no] lsp-refresh-interval <secs>

The <secs> parameter specifies the maximum refresh interval and can be from 1 – 65535 seconds. The default is 900 seconds (15 minutes).

## Changing the LSP General Interval

The LSP general interval is the minimum number of seconds the Layer 3 Switch waits between sending updated LSPs to its IS-IS neighbors. The interval can be from 1 – 120 seconds. The default is 10 seconds.

To change the LSP general interval to 45 seconds, enter a command such as the following:

```
BigIron MG8(config-isis-router)# lsp-gen-interval 45
```

***Syntax:*** [no] lsp-gen-interval <secs>

The <secs> parameter specifies the minimum refresh interval and can be from 1 – 120 seconds. The default is 10 seconds.

## Changing the SPF Timer

Every IS maintains a Shortest Path First (SPF) tree, which is a representation of the states of each of the IS's links to ESs and other ISs. If the IS is both a Level-1 and Level-2 IS, it maintains separate SPF trees for each level.

To ensure that the SPF tree remains current, the IS updates the tree at regular intervals following a change in network topology or the link state database. By default, the Foundry Layer 3 Switch recalculates its IS-IS tree every five seconds following a change. You can change the SPF timer to a value from 1 – 120 seconds.

To change the SPF interval, enter a command such as the following:

```
BigIron MG8(config-isis-router)# spf-interval 30
```

***Syntax:*** [no] spf-interval <secs>

The <secs> parameter specifies the interval and can be from 1 – 120 seconds. The default is 5 seconds.

## Globally Disabling or Re-Enabling Hello Padding

By default, the Layer 3 Switch adds extra data to the end of a hello packet to make the packet the same size as the maximum length of PDU the Layer 3 Switch supports.

The padding applies to the following types of hello packets:

- ES hello (ESH PDU)
- IS hello (ISH PDU)
- IS to IS hello (IIH PDU)

The padding consists of arbitrarily valued octets. A padded hello PDU indicates the largest PDU that the Layer 3 Switch can receive. Other ISs that receive a padded hello PDU from the Layer 3 Switch can therefore ensure that the IS-IS PDUs they send the Layer 3 Switch. Similarly, if the Layer 3 Switch receives a padded hello PDU from a neighbor IS, the Layer 3 Switch knows the maximum size PDU that the Layer 3 Switch can send to the neighbor.

When padding is enabled, the maximum length of a Hello PDU sent by the Layer 3 Switch is 1514 bytes.

If you need to disable padding, you can do so globally or on individual interfaces. Generally, you do not need to disable padding unless a link is experiencing slow performance, for example due to point-to-point interoperability issues. If you enable or disable padding on an interface, the interface setting overrides the global setting.

By default, disabling or re-enabling padding affects hello PDUs sent on point-to-point circuits and to an IS-IS broadcast address. You can specify an option to enable or disable the padding for point-to-point or broadcast PDUs.

To globally disable padding of IS-IS hello PDUs, enter the following command:

```
NetIron(config-isis-router)# no hello padding
```

This command disables all hello PDU padding on the Layer 3 Switch. To re-enable padding, enter the following command:

```
BigIron MG8(config-isis-router)# hello padding
```

To disable padding on a specific interface only, enter commands such as the following:

```
BigIron MG8(config-isis-router)# interface ethernet 1/1
BigIron MG8(config-if-1/1)# hello padding
```

***Syntax:*** [no] hello padding [point-to-point]

The **point-to-point** parameter disables or re-enables the padding only for point-to-point connections.

Enter the **no** form of the command to re-enable hello padding.

To disable hello padding, see “Disabling and Enabling Hello Padding on an Interface” on page 7-21.

### Logging Adjacency Changes

The Layer 3 Switch can generate a Syslog entry and an SNMP trap to indicate a change in the status of an adjacency with another IS. Logging of the adjacency changes is disabled by default. To enable or disable them, use either of the following methods.

To display the Syslog messages, see *Foundry Switch and Router Installation and Basic Configuration Guide*.

To enable logging of adjacency changes, enter the following command:

```
NetIron(config-isis-router)# log-adjacency-changes
```

***Syntax:*** [no] log-adjacency-changes

To disable logging of adjacency changes, enter the following command:

```
NetIron(config-isis-router)# no log-adjacency-changes
```

### Disabling Partial SPF Calculations

**NOTE:** This feature is not supported on Terathon devices.

By default, IS-IS makes incremental changes to the routing table when changes to the network occur. A full SPF calculation is not performed unless there is a substantial change in the network; for example when an IS-IS link flaps in the network. You can optionally configure IS-IS to perform a full SPF calculation when any changes occur in the network.

To disable partial SPF calculations for IS-IS, enter the following command:

```
BigIron MG8(config-isis-router-ipv6u)# disable-partial-spf-opt
```

***Syntax:*** [no] disable-partial-spf-opt

This command applies to both IPv4 and IPv6 address families, if both are configured.

## Configuring IPv6 Address Family Route Parameters

This section describes how to modify the IS-IS the parameters for the IS-IS IPv6 address family.

### Changing the Maximum Number of Load Sharing Paths

By default, IPv6 IS-IS can calculate and install four equal-cost paths into the IPv6 forwarding table. You can change the number of paths IPv6 IS-IS can calculate and install in the IPv6 forwarding table to an amount from 1 – 8. If you change the number of paths to one, the Foundry device does not load share route paths learned from IPv6 IS-IS.

For example, to change the number of paths IPv6 IS-IS can calculate and install in the IPv6 forwarding table to three, enter the following command at the IPv6 IS-IS unicast address family configuration level:

```
BigIron MG8(config-isis-router-ipv6u)# maximum-paths 8
```

***Syntax:*** [no] maximum-paths <number>

The <number> parameter specifies the number of paths IPv6 IS-IS can calculate and install in the IPv6 forwarding table.

To return to the default number of maximum paths, enter the **no** form of this command.

## Enabling Advertisement of a Default Route

By default, the Foundry device does not generate or advertise a default route to its neighboring ISs. A default route is not advertised even if the device's IPv6 route table contains a default route. You can enable the device to advertise a default route to all neighboring ISs using one of the following methods. By default, the feature originates the default route at Level 2 only. However, you can apply a route map to originate the default route to Level 1 only or at both Level 1 and Level 2.

**NOTE:** This feature requires the presence of a default route in the IPv6 route table.

To enable the Foundry device to advertise a default route that is originated a Level 2, enter the following command at the IPv6 IS-IS unicast address family configuration level:

```
BigIron MG8(config-isis-router-ipv6u)# default-information-originate
```

This command enables the device to advertise a default route into the IPv6 IS-IS area to which the device is attached.

***Syntax:*** [no] default-information-originate [route-map <name>]

The **route-map** <name> parameter allows you to specify the level on which to advertise the default route. You can specify one of the following:

- Advertise to Level-1 ISs only.
- Advertise to Level-2 ISs only.
- Advertise to Level-1 and Level-2 ISs.

**NOTE:** The route map must be configured before you can use the route map as a parameter with the **default-information-originate** command.

To use a route map to specify the router to advertise a default route to Level 1, enter commands such as the following at the Global CONFIG level:

```
BigIron MG8(config)# route-map default_level1 permit 1
BigIron MG8(config-routemap default_level1)# set level level-1
BigIron MG8(config-routemap default_level1)# router isis
BigIron MG8(config-isis-router)# address-family ipv6 unicast
BigIron MG8(config-isis-router-ipv6u)# default-information-originate route-map
default_level1
```

These commands configure a route map to set the default advertisement level to Level 1 only.

***Syntax:*** [no] route-map <map-name> permit | deny <sequence-number>

***Syntax:*** [no] set level level-1 | level-1-2 | level-2

For this use of a route map, use the **permit** option and do not specify a **match** statement. Specify a **set** statement to set the level to one of the following:

- **level-1** – Level 1 only.
- **level-1-2** – Level 1 and Level 2.
- **level-2** – Level 2 only (default).

## Changing the Administrative Distance for IPv6 IS-IS

When the Foundry device has paths from multiple routing protocols to the same destination, it compares the administrative distances of the paths and selects the path with the lowest administrative distance to place in the IPv6 route table.

For example, if the router has a path from RIPng, from OSPFv3, and IPv6 IS-IS to the same destination, and all the paths are using their protocols' default administrative distances, the router selects the OSPFv3 path, because that path has a lower administrative distance than the RIPng and IPv6 IS-IS paths.

Here are the default IPv6 administrative distances on the Foundry device:

- Directly connected – 0 (this value is not configurable)
- Static – 1 (applies to all static routes, including default routes)
- EBGP – 20
- OSPFv3 – 110
- IPv6 IS-IS – 115
- RIPng – 120
- IBGP – 200
- Local BGP – 200
- Unknown – 255 (the device will not use this route)

Lower administrative distances are preferred over higher distances. For example, if the Foundry device receives routes for the same network from IPv6 IS-IS and from RIPng, it will prefer the IPv6 IS-IS route by default.

To change the administrative distance for IPv6 IS-IS routes, enter the following command at the IPv6 IS-IS unicast address family configuration level:

```
BigIron MG8(config-isis-router-ipv6u)# distance 100
```

***Syntax:*** [no] distance <number>

This command changes the administrative distance for all IPv6 IS-IS routes to 100.

The <number> parameter specifies the administrative distance. You can specify a value from 1 – 255. (Routes with a distance value of 255 are not installed in the routing table.) The default for IPv6 IS-IS is 115.

## Configuring Summary Prefixes

You can configure summary prefixes to aggregate IPv6 IS-IS route information. Summary prefixes can enhance performance by reducing the size of the Link State database, reducing the amount of data a router needs to send to its neighbors, and reducing the CPU cycles used for IPv6 IS-IS.

When you configure a summary prefix, the prefix applies only to Level-2 routes by default. You can specify Level-1 only, Level-2 only, or Level-1 and Level-2 when you configure the prefix.

For example, to configure a summary prefix of 2001:e0ff::/32 to be advertised to Level-1 routes only, enter the following command at the IPv6 IS-IS unicast address family configuration level:

```
BigIron MG8(config-isis-router-ipv6u)# summary-address 2001:e0ff::/32 level-1
```

***Syntax:*** [no] summary-address <ipv6-prefix>/<prefix-length> [level-1 | level-1-2 | level-2-only]

The <ipv6-prefix>/<prefix-length> parameter specifies the aggregate address. You must specify the <ipv6-prefix> parameter in hexadecimal using 16-bit values between colons as documented in RFC 2373. You must specify the <prefix-length> parameter as a decimal value. A slash mark (/) must follow the <ipv6-prefix> parameter and precede the <prefix-length> parameter.

The **level-1** | **level-1-2** | **level-2-only** parameter specifies the route types to which the aggregate route applies. The default is **level-2-only**.

## Redistributing Routes into IPv6 IS-IS

To redistribute routes into IPv6 IS-IS, you can perform the following configuration tasks:

- Change the default redistribution metric (optional).
- Configure the redistribution of a particular route type into IPv6 IS-IS (mandatory).

The Foundry device can redistribute routes from the following route sources into IPv6 IS-IS:

- BGP4+.
- RIPng.
- OSPFv3.
- Static IPv6 routes.
- IPv6 routes learned from directly connected networks.

The Foundry device can also can redistribute Level-1 IPv6 IS-IS routes into Level-2 IPv6 IS-IS routes, and Level-2 IPv6 IS-IS routes into Level-1 IPv6 IS-IS routes.

Route redistribution from other sources into IPv6 IS-IS is disabled by default. When you enable redistribution, the device redistributes routes only into Level 2 by default. You can specify Level 1 only, Level 2 only, or Level 1 and Level 2 when you enable redistribution.

The device automatically redistributes Level-1 routes into Level-2 routes. Thus, you do not need to enable this type of redistribution. You also can enable redistribution of Level-2 routes into Level-1 routes.

The device attempts to use the redistributed route's metric as the route's IPv6 IS-IS metric. For example, if an OSPFv3 route has an OSPF cost of 20, the router uses 20 as the route's IPv6 IS-IS metric. The device uses the redistributed route's metric as the IPv6 IS-IS metric unless the route does not a have a valid metric. In this case, the device assigns the default metric value to the route. For information about the default metric, see the "Changing the Default Redistribution Metric" section, which follows this section.

## Changing the Default Redistribution Metric

When IPv6 IS-IS redistributes a route from another route source (such as OSPFv3, BGP4+, or a static IPv6 route) into IPv6 IS-IS, it uses the route's metric value as its metric when the metric is not modified by a route map or metric parameter and the default redistribution metric is set to its default value of 0. You can change the default metric to a value from 0 – 65535.

---

**NOTE:** The Foundry implementation of IS-IS does not support the optional metric types Delay, Expense, or Error.

---

For example, to change the default metric to 20, enter the following command at the IPv6 IS-IS unicast address family configuration level:

```
BigIron MG8(config-isis-router-ipv6u)# default-metric 20
```

***Syntax:*** [no] default-metric <number>

The <number> parameter specifies the default metric. You can specify a value from 1 – 65535. The default is 10.

The restore the default value for the default metric, enter the **no** form of this command.

## Redistributing Static IPv6 Routes into IPv6 IS-IS

To redistribute static IPv6 routes from the IPv6 static route table into IPv6 IS-IS routes, enter the following command at the IPv6 IS-IS unicast address family configuration level:

```
BigIron MG8(config-isis-router-ipv6u)# redistribute static
```

This command configures the Foundry device to redistribute all static IPv6 routes into Level-2 IS-IS routes.

***Syntax:*** [no] redistribute static [level-1 | level-1-2 | level-2 | metric <number> | metric-type external | internal | route-map <name>]

The **level-1**, **level-1-2**, and **level-2** keywords restrict redistribution to the specified IPv6 IS-IS level.

The **metric** <number> parameter restricts the redistribution to only those routes that have the metric you specify.

The **metric-type external** | **internal** parameter restricts redistribution to one of the following:

- **external** – The metric value is not comparable to an IPv6 IS-IS internal metric and is always higher than the IPv6 IS-IS internal metric.
- **internal** – The metric value is comparable to metric values used by IPv6 IS-IS. This is the default.

The **route-map** <name> parameter restricts redistribution to those routes that match the specified route map. The route map must already be configured before you use the route map name with the **redistribute** command. For example, to configure a route map that redistributes only the static IPv6 routes to the destination networks 2001:100::/32, enter commands such as the following:

```
BigIron MG8(config)# ipv6 access-list static permit any 2001:100::/32
BigIron MG8(config)# route-map static permit 1
BigIron MG8(config-routemap static)# match ip address static
BigIron MG8(config-routemap static)# router isis
BigIron MG8(config-isis-router)# address-family ipv6 unicast
BigIron MG8(config-isis-router-ipv6u)# redistribute static route-map static
```

For information about the IPv6 ACL and route map syntax, see the following:

- “Configuring an IPv6 Access Control List” on page 10-1.
- The “Defining Route Maps” section in the “Configuring BGP4” chapter of the *Foundry NetIron Service Provider Configuration and Management Guide*.

## Redistributing Directly Connected Routes into IPv6 IS-IS

To redistribute directly connected IPv6 routes into IPv6 IS-IS routes, enter the following command at the IPv6 IS-IS unicast address family configuration level:

```
BigIron MG8(config-isis-router-ipv6u)# redistribute connected
```

This command configures the Foundry device  to redistribute all directly connected routes in the IPv6 route table into Level-2 IPv6 IS-IS.

***Syntax:*** [no] redistribute connected [level-1 | level-1-2 | level-2 | metric <number> | metric-type external | internal | route-map <name>]

The parameters are the same as the parameters for the **redistribute static** command.

## Redistributing RIPng Routes into IPv6 IS-IS

To redistribute RIPng routes into IPv6 IS-IS, enter the following command at the IPv6 IS-IS unicast address family configuration level:

```
BigIron MG8(config-isis-router-ipv6u)# redistribute rip
```

This command configures the Foundry device to redistribute all RIPng routes into Level-2 IS-IS.

***Syntax:*** [no] redistribute rip [level-1 | level-1-2 | level-2 | metric <number> | metric-type external | internal | route-map <name>]

The parameters are the same as the parameters for the **redistribute static** command.

## Redistributing OSPF Version 3 Routes into IPv6 IS-IS

To redistribute OSPFv3 routes into IPv6 IS-IS, enter the following command at the IPv6 IS-IS unicast address family configuration level:

```
BigIron MG8(config-isis-router-ipv6u)# redistribute ospf
```

This command configures the Foundry device to redistribute all OSPFv3 routes into Level-2 IPv6 IS-IS.

***Syntax:*** [no] redistribute ospf [level-1 | level-1-2 | level-2 | match external1 | external2 | internal | metric <number> | metric-type external | internal | route-map <name>]

Most of the parameters are the same as the parameters for the **redistribute static** command. However, the **redistribute ospf** command also has the **match external1** | **external2** | **internal** parameter. This parameter specifies the OSPF route type you want to redistribute into IPv6 IS-IS. By default, the **redistribute ospf** command redistributes only internal routes.

- **external1** – An OSPF type 1 external route.
- **external2** – An OSPF type 2 external route.
- **internal** – An internal route calculated by OSPF.

## Redistributing BGP4+ Routes into IPv6 IS-IS

To redistribute BGP4+ routes into IPv6 IS-IS, enter the following command at the IPv6 IS-IS unicast address family configuration level:

```
BigIron MG8(config-isis-router-ipv6u)# redistribute bgp
```

This command configures the router to redistribute all its BGP4 routes into Level-2 IPv6 IS-IS.

***Syntax:*** [no] redistribute bgp [level-1 | level-1-2 | level-2 | metric <number> | metric-type external | internal | route-map <name>]

The parameters are the same as the parameters for the **redistribute static** command.

## Redistributing IPv6 IS-IS Routes Within IPv6 IS-IS

In addition to redistributing routes from other route sources into IPv6 IS-IS, the Foundry device can redistribute Level 1 IPv6 IS-IS routes into Level 2 IPv6 IS-IS routes, and Level 2 IPv6 IS-IS routes into Level 1 IPv6 IS-IS routes. By default, the device redistributes routes from Level 1 into Level 2.

---

**NOTE:** The Foundry device automatically redistributes Level 1 routes into Level 2 routes, even if you do not enable redistribution.

---

For example, to redistribute all IPv6 IS-IS routes from Level 2 into Level 1, enter the following command at the IPv6 IS-IS unicast address family configuration level:

```
BigIron MG8(config-isis-router-ipv6u)# redistribute isis level-2 into level-1
```

The router automatically redistributes Level-1 routes into Level 2.

***Syntax:*** [no] redistribute isis level-1 into level-2 | level-2 into level-1 [prefix-list <name>]

The **level-1 into level-2** | **level-2 into level-1** parameter specifies the direction of the redistribution:

- **level-1 into level-2** – Redistributes Level 1 routes into Level 2. This is the default.
- **level-2 into level-1** – Redistributes Level 2 routes into Level 1.

The optional **prefix-list** <name> parameter allows you to specify the IPv6 prefixes that you want redistributed from Level 1 into Level 2 and from Level 2 into Level 1. Specify the name of the IPv6 prefix list that contains the desired prefixes. (For information about prefix lists, including the syntax of the **ipv6 prefix-list** command, see “Configuring an IPv6 Prefix List” on page 11-1.)

For example, to redistribute the IPv6 prefix 2001::/16 from Level 2 into Level 1, enter commands such as the following:

```
BigIron MG8(config)# ipv6 prefix-list routesfor2001 permit 2001::/16
BigIron MG8(config)# router isis
BigIron MG8(config-isis-router)# address-family ipv6 unicast
BigIron MG8(config-isis-router-ipv6u)# redistribute isis level-2 into level-1
```

```
prefix-list routesfor2001
```

### Disabling and Reenabling IPv6 Protocol-Support Consistency Checks

As discussed in "IPv6 IS-IS Single-Topology Mode" on page 7-5, an IS-IS single topology must be configured to run the same set of network protocols (IPv4 IS-IS only, IPv6 IS-IS only, or both IPv4 IS-IS and IPv6 IS-IS).

By default, IS-IS performs consistency checks on hello packets. If a hello packet does not have the same configured network protocols, IS-IS rejects the packet. For example, a hello packet from a router running IPv4 and IPv6 IS-IS will be rejected by a router running either IPv4 IS-IS only or IPv6 IS-IS only, and the two routers will not become adjacent.

To allow two routers running different sets of network protocols to form an adjacency, enter the following command at the IPv6 IS-IS unicast address family configuration level:

```
BigIron MG8(config-isis-router-ipv6u)# no adjacency-check
```

This command disables the IPv6 IS-IS consistency check.

***Syntax:*** [no] adjacency-check

To reenable the consistency check, enter the following command at the IPv6 IS-IS unicast address family configuration level:

```
BigIron MG8(config-isis-router-ipv6u)# adjacency-check
```

## Configuring ISIS Properties on an Interface

This section describe the IS-IS parameters for an interface.

### Disabling or Re-Enabling Formation of Adjacencies

When you enable IS-IS on any type of interface except a loopback interface, the interface also is enabled to send advertisements and form an adjacency with an IS at the other end of the link by default. Adjacency formation and advertisements are disabled by default on loopback interfaces.

You can enable or disable adjacency formation and advertisements on an interface.

---

**NOTE:** The Foundry device advertises an IS-IS interface to its area regardless of whether adjacency formation is enabled.

---

To disable IS-IS adjacency formation on an interface, enter commands such as the following:

```
BigIron MG8(config-isis-router)# interface ethernet 2/8
BigIron MG8(config-if-e1000-2/8)# isis passive
```

This command disables IS-IS adjacency formation on port 2/8. The device still advertises this IS-IS interface into the area, but does not allow the port to form an adjacency with the IS at the other end of the link.

***Syntax:*** [no] isis passive

### Setting the Priority for Designated IS Election

The priority of an IS-IS interface determines the priority of the interface for being elected as a Designated IS. Level-1 has a Designated IS and Level-2 has a Designated IS. The Level-1 and Level-2 Designated ISs are independent, although the same device can become both the Level-1 Designated IS and the Level-2 Designated IS.

By default, the Level-1 and Level-2 priority is 64. You can configure an interface's priority to a value from 0 – 127. You can configure the same priority for both Level-1 and Level-2 or you can configure a different priority for each level. In case of a tie (if two or more devices have the highest priority within a given level), the device with the highest MAC address becomes the Designated IS for that level.

**NOTE:** You can set the IS-IS priority on an individual interface basis only. You cannot set the priority globally.

To set the IS-IS priority on an interface, use either of the following methods.

To set the IS-IS priority on an interface, enter commands such as the following:

```
BigIron MG8(config-isis-router)# interface ethernet 2/8
BigIron MG8(config-if-e1000-2/8)# isis priority 127
```

This command sets the IS-IS priority on port 1/1 to 127. Since the command does not specify Level-1 or Level-2, the new priority setting applies to both IS-IS levels.

***Syntax:*** [no] isis priority <num> [level-1-only | level-2-only]

The <num> parameter specifies the priority and can be from 0 – 127. A higher numeric value means a higher priority. The default is 64.

The **level-1-only | level-2-only** parameter applies the priority to Level-1 only or Level-2 only. By default, the priority is applied to both levels.

## Limiting Access to Adjacencies With a Neighbor

Instead of limiting access to an area (level-1) or domain (level-2) you can limit access in forming a connection on a per interface/circuit level by entering a password at the interface level. To enter this password, enter a command such as the following:

```
BigIron MG8(config-isis-router)# interface ethernet 2/8
BigIron MG8(config-if-e1000-2/8)# isis password
```

***Syntax:*** [no] isis password

## Changing the IS-IS Level on an Interface

The section “Changing the IS-IS Level Globally” on page 7-11 explains how to change the IS-IS level globally. By default, a Foundry Layer 3 Switch can operate as both a Level-1 and IS-IS Level-2 router. You can change the IS-IS type on an individual interface to be Level-1 only or Level-2 only. You also can reset the type to both Level-1 and Level-2.

**NOTE:** If you change the IS-IS type on an individual interface, the type you specify must also be specified globally. For example, if you globally set the type to Level-2 only, you cannot set the type on an individual interface to Level-1. The software accepts the setting but the setting does not take effect.

To change the IS-IS type on a specific interface, enter commands such as the following:

```
BigIron MG8(config-isis-router)# interface ethernet 2/8
BigIron MG8(config-if-e1000-2/8)# isis circuit-type level-1
```

***Syntax:*** [no] isis circuit-type level-1 | level-1-2 | level-2

The **level-1-only | level-1-2 | level-2-only** parameter specifies the IS-IS type. If you want to re-enable support for both IS-IS types, re-enter the command you entered to change the IS-IS type, and use “no” in front of the command.

## Disabling and Enabling Hello Padding on an Interface

The section “Globally Disabling or Re-Enabling Hello Padding” on page 7-13 explains what hello padding is, why it is important and how to globally disable or enable it on a device. You can also disable hello padding on a specific interface by entering commands such as the following:

```
BigIron MG8(config-isis-router)# interface ethernet 2/8
BigIron MG8(config-if-e1000-2/8)# hello padding
```

***Syntax:*** [no] hello padding [point-to-point]

The **point-to-point** parameter disables or re-enables the padding only for point-to-point connections.

Enter the **no** form of the command to re-enable hello padding.

## Changing the Hello Interval

The hello interval controls how often an IS-IS interface sends hello messages to its IS-IS neighbors. The default interval is 10 seconds for Level-1 and Level-2. You can change the hello interval for one or both levels to a value from 1 – 65535 seconds.

To change the hello interval for Ethernet interface 2/8, enter commands such as the following:

```
BigIron MG8(config-isis-router)# interface ethernet 2/8
BigIron MG8(config-if-e1000-2/8)# isis hello-interval 20
```

This command changes the hello interval to 20 seconds. By default, the change applies to both Level-1 and Level-2.

***Syntax:*** [no] isis hello-interval <num> [level-1-only | level-2-only]

The <num> parameter specifies the interval, and can be from 1 – 65535 seconds. The default is 10 seconds.

The **level-1-only | level-2-only** parameter applies the change to only the level you specify. If you do not use this parameter, the change applies to both levels.

## Changing the Hello Multiplier

The hello multiplier is the number by which an IS-IS interface multiplies the hello interval to obtain the hold time for Level-1 and Level-2 IS-to-IS hello PDUs. The default multiplier is 3. You can set the multiplier to a value from 3 – 1000.

To change the hello multiplier for Ethernet interface 2/8, enter commands such as the following:

```
BigIron MG8(config-isis-router)# interface ethernet 2/8
BigIron MG8(config-if-e1000-2/8)# isis hello-multiplier 50
```

This command changes the hello interval to 50. By default, the change applies to both Level-1 and Level-2.

***Syntax:*** [no] isis hello-multiplier <num> [level-1-only | level-2-only]

The <num> parameter specifies the multiplier, and can be from 3 – 1000. The default is 3.

The **level-1-only | level-2-only** parameter applies the change to only the level you specify. If you do not use this parameter, the change applies to both levels.

## Changing the Metric Added to Advertised Routes

When the Layer 3 Switch originates an IS-IS route or calculates a route, the Layer 3 Switch adds a metric (cost) to the route. Each IS-IS interface has a separate metric value. The default is 10.

The Layer 3 Switch applies the interface-level metric to routes originated on the interface and also when calculating routes. The Layer 3 Switch does not apply the metric to link-state information that the Layer 3 Switch receives from one IS and floods to other ISs.

The default interface metric is 10. You can change the metric on an individual interface to a value in one of the following ranges:

- 1 – 63 for the narrow metric style (the default metric style for IPv6 ISIS)
- 1 – 16777215 for the wide metric style (the default metric style for IPv4 ISIS)

**NOTE:** If the metric value you want to use is higher than 63 but you have not changed the metric style to wide, change the metric style first, then set the metric. The IS-IS neighbors that will receive the advertisements also must be enabled to receive wide metrics.

To change the IS-IS metric on an interface, use the following CLI method.

```
BigIron MG8(config-isis-router)# interface ethernet 2/8
BigIron MG8(config-if-e1000-2/8)# isis metric
```

***Syntax:*** [no] isis metric <num>

The <num> parameter specifies the metric. The range of values you can specify depends on the metric style. You can specify 1 – 63 for the narrow metric style or 1 – 16777215 for the wide metric style. The default in either case is 10.

When IPv6 IS-IS is enabled in a single topology, you must set the value of the metric for a wide style. If both IPv4 IS-IS and IPv6 IS-IS are configured on an interface, the value of the metric must be for a wide style for both network protocols on all interfaces.

# Displaying IPv6 IS-IS Information

You can display the following information about IPv6 IS-IS:

- General IPv6 IS-IS information.
- IPv6 IS-IS configuration information.
- IPv6 IS-IS error statistics.
- LSP database entries.
- IS-IS system ID to hostname mappings.
- IPv6 IS-IS interface information.
- IPv6 IS-IS memory usage information.
- IPv6 IS-IS neighbor information.
- IPv6 IS-IS path information.
- IPv6 IS-IS redistribution information.
- IPv6 IS-IS route information.
- IPv6 IS-IS traffic statistics.

## Displaying IPv6 IS-IS Information

To display general IPv6 IS-IS information, enter the following command at any CLI level:

```
BigIron MG8# show ipv6 isis
 IS-IS  Routing Protocol Operation State: Enabled
 IS-Type: Level-1-2
 System ID: 8888.5555.0008
 Manual area address(es):
   49.8585
 Interfaces with Integrated IS-IS for IPv6 configured:
    Interface 4/1    Interface 4/2    Interface 4/11    Interface 4/12
    Interface 4/13   Interface 4/14   Interface 4/15    Interface 4/16
    Interface 4/17   Interface 4/35   Interface 4/36    Interface 4/37
    Interface 4/38   Interface v43    Interface v44     Interface lb1
 Following Routes are Redistributed into IS-IS for IPv6:
   CONNECTED
 Number of Routes redistributed into IS-IS: 1
 Domain password: None
 Area password: None
 IS-IS for IPV6 Route Administrative Distance: 115
 Hold Time Between Two SPF Calculations: 5
 Global Hello Padding: Enabled
 Global Hello Padding For Point to Point Circuits: Enabled
```

***Syntax:*** show ipv6 isis

This display shows the following information:

**Table 7.2: IPv6 IS-IS information fields**

| This Field... | Displays... |
|---|---|
| IS-IS Routing Protocol Operation State | The operating state of IPv6 IS-IS. Possible states include the following:<br>• Enabled – IPv6 IS-IS is enabled.<br>• Disabled – IPv6 IS-IS is disabled. |
| IS Type | The intermediate system type. Possible types include the following:<br>• Level 1 only – The Foundry device routes traffic only within the area in which it resides.<br>• Level 2 only – The Foundry device routes traffic between areas of a routing domain.<br>• Level 1-2 – The Foundry device routes traffic within the area in which it resides and between areas of a routing domain. |
| System ID | The unique IS-IS router ID. Typically, the Foundry device's base MAC address is used as the system ID. |
| Manual area address(es) | Area address(es) of the Foundry device. |
| Interfaces with Integrated IS-IS for IPv6 configured | Interfaces on which IPv6 IS-IS is configured. |
| Following Routes are Redistributed into IS-IS for IPv6 | Routes that are redistributed into IPv6 IS-IS. Possible routes include the following:<br>• BGP – BGP4+ routes are redistributed into IPv6 IS-IS.<br>• RIP – RIPng routes are redistributed into IPv6 IS-IS.<br>• OSPF – OSPFv3 routes are redistributed into IPv6 IS-IS.<br>• STATIC – Static IPv6 routes are redistributed into IPv6 IS-IS.<br>• CONNECTED – IPv6 routes learned from directly connected networks are redistributed into IPv6 IS-IS. |
| Number of Routes redistributed into IS-IS | The number of routes distributed into IS-IS |
| Domain password | The domain password, if one is configured. |
| Area password | The domain password, if one is configured. |
| IS-IS IPv6 Route Administrative Distance | The current setting of the IPv6 IS-IS administrative distance. |
| Hold Time Between Two SPF Calculations | The setting of the SPF timer, which causes the router to recalculate the SPF tree of its IPv6 IS-IS links following a change in topology or the link state database. |
| Global Hello Padding | The setting of the global hello padding feature, which can be one of the following:<br>• Disabled – Global padding for hello packets is disabled.<br>• Enabled – Global padding for hello packets is enabled. |

**Table 7.2: IPv6 IS-IS information fields(Continued)**

| This Field... | Displays... |
|---|---|
| Global Hello Padding for Point to Point Circuits | The setting of the hello padding feature on point-to-point circuits and to IPv6 IS-IS broadcast addresses, which can be one of the following:<br>• Disabled – Padding for hello packets on point-to-point circuits and to IPv6 IS-IS broadcast addresses is disabled.<br>• Enabled – Padding for hello packets on point-to-point circuits and to IPv6 IS-IS broadcast addresses is enabled. |

## Displaying the IPv6 IS-IS Configuration in the Running Configuration

You can display the IPv6 IS-IS commands that are in effect on the Foundry device.

**NOTE:** The running configuration does not list the default values. Only commands that change a setting or add configuration information are displayed.

To display the IPv6 IS-IS configuration, enter the following command at any CLI level:

```
BigIron MG8# show ipv6 isis config
Current ISIS configuration:
router isis
 net  49.6561.2222.2222.2222.00

 address-family ipv4 unicast
 distance  135
 redistribute static
 exit-address-family

 address-family ipv6 unicast
 redistribute static
 exit-address-family

end
```

***Syntax:*** show ipv6 isis config

The running configuration shown in this example contains the following commands:

- Global IPv6 IS-IS commands that enable IS-IS.
- Address family commands that configure IPv4 IS-IS unicast routes.
- Address family commands that configure IPv6 IS-IS unicast routes.

## Displaying IPv6 IS-IS Error Statistics

To display IPv6 IS-IS error statistics, enter the following command at any level of the CLI:

```
BigIron MG8# show ipv6 isis counts
Area Mismatch: 0
Max Area Mismatch: 0
System ID Length Mismatch: 0
Authentication Fail: 0
Corrupted LSP: 0
LSP Sequence Number Skipped: 0
LSP Max Sequence Number Exceeded: 0
Level-1 Database Overload: 0
Level-2 Database Overload: 0
Our LSP Purged: 0
```

***Syntax:*** show ipv6 isis counts

This display shows the following information:

**Table 7.3: IPv6 IS-IS error statistics**

| This Field... | Displays... |
|---|---|
| Area Mismatch | The number of times the router interface was unable to create a Level-1 adjacency with a neighbor because the router interface and the neighbor did not have any areas in common. |
| Max Area Mismatch | The number of times the Foundry device received a PDU with a value for maximum number of area addresses that did not match the device's value for maximum number of area addresses. |
| System ID Length Mismatch | The number of times the Foundry device received a PDU with an ID field that was a different length than the ID field length configured on the router. |
| Authentication Fail | The number of times authentication failed because the Foundry device was configured to authenticate IPv6 IS-IS packets in the packet's domain or area, but the packet did not contain the correct password. |
| Corrupted LSP | The number of times the Foundry device detected a corrupted LSP in the device's memory. |
| LSP Sequence Number Skipped | The number of times the device received an LSP with a sequence number that was more than 1 higher than the sequence number of the previous LSP received from the same neighbor. |
| LSP Max Sequence Number Exceeded | The number of times the device attempted to set an LSP sequence number to a value higher than the highest number in the CSNP sent by the Designated IS. |

**Table 7.3: IPv6 IS-IS error statistics (Continued)**

| This Field... | Displays... |
|---|---|
| Level-1 Database Overload | The number of times the Level-1 state on the router changed from Waiting to On or from On to Waiting.<br>• Waiting to On – This change can occur when the device recovers from a previous Level-1 LSP database overload and is again ready to receive new LSPs.<br>• On to Waiting – This change can occur when the device's Level-1 LSP database is full and the device receives an additional LSP, for which there is no room. |
| Level-2 Database Overload | The number of times the Level-2 state on the device changed from Waiting to On or from On to Waiting.<br>• The change from Waiting to On can occur when the device recovers from a previous Level-2 LSP database overload and is again ready to receive new LSPs.<br>• The change from On to Waiting can occur when the device's Level-2 LSP database is full and the device receives an additional LSP, for which there is no room. |
| Our LSP Purged | The number of times the device received an LSP that was originated by the device itself and had age zero (aged out). |

## Displaying LSP Database Entries

You can display summary or detailed information about the entries in the LSP database.

**NOTE:** The router maintains separate LSP databases for Level 1 LSPs and Level 2 LSPs.

To display summary information about the entries in the LSP database, enter the following command at any level of the CLI:

```
BigIron MG8# show ipv6 isis database
IS-IS Level-1 Link State Database
LSPID                 LSP Seq Num  LSP Checksum  LSP Holdtime      ATT/P/OL
Router1.00-00         0x00000003   0x9a6b        574               0/0/0
Router2.00-00*        0x00000002   0x609d        540               0/0/0
Router2.01-00*        0x00000001   0x0fcf        539               0/0/0

IS-IS Level-2 Link State Database
LSPID                 LSP Seq Num  LSP Checksum  LSP Holdtime      ATT/P/OL
Router1.00-00         0x00000003   0xe2da        574               0/0/0
Router2.00-00*        0x00000002   0x0585        540               0/0/0
Router2.01-00*        0x00000001   0x0fcf        539               0/0/0
```

The command in this example displays information for the LSPS in the router's Level-1 and Level-2 LSP databases. Notice that the display groups the Level-1 and Level-2 LSPs separately.

***Syntax:*** show ipv6 isis database [<HHHH.HHHH.HHHH.HH-HH> | detail | l1 | l2 | level1 | level2]

The <HHHH.HHHH.HHHH.HH-HH> parameter restricts the display to the entry for the specified LSPID. (The LSPID consists of the source ID (HHHH.HHHH.HHHH), the pseudonode (HH-), and LSPID (-HH). To determine

the router's source ID, use the **show ipv6 isis** command. For more information, see "Displaying IPv6 IS-IS Information" on page 7-23. To determine the pseudonode and LSPID, use the **show ipv6 isis database** command.

**NOTE:** Name mapping is enabled by default. When name mapping is enabled, the output of the **show ipv6 isis database** command uses the hostname instead of the system ID. To disable mapping so that these displays use the system ID instead, enter the **no hostname** command at the IS-IS router configuration level. For more information about performing this task, see the "Configuring IS-IS" chapter in the *Foundry NetIron Service Provider Configuration and Management Guide*.

The **detail** parameter displays detailed information about the LSPs. The detailed information display is discussed later in this section.

The **l1** and **level1** parameters restrict the display to the Level-1 LSP entries. You can use these parameters interchangeably.

The **l2** and **level2** parameters restrict the display to the Level-2 LSP entries. You can use these parameters interchangeably.

This display shows the following information:

**Table 7.4: IPv6 IS-IS summary LSP database information**

| This Field... | Displays... |
|---|---|
| LSPID | The LSP ID, which consists of the source ID (HHHH.HHHH.HHHH), the pseudonode (HH-), and LSPID (-HH).<br>**Note**: If the address has an asterisk ( * ) at the end, this indicates that the LSP is locally originated. |
| LSP Seq Num | The sequence number of the LSP. |
| LSP Checksum | The checksum calculated by the device that sent the LSP and used by the Foundry device to verify that the LSP was not corrupted during transmission over the network. |
| LSP Holdtime | The maximum number of seconds during which the LSP will remain valid.<br>**Note**: The IS that originates the LSP starts the timer for the LSP. As a result, LSPs do not all have the same amount of time remaining when they enter the Foundry device's LSP database. |
| ATT | A 4-bit value extracted from bits 4 – 7 in the Attach field of the LSP. |
| P | The value in the Partition option field of the LSP. The field can have one of the following values:<br>• 0 – The IS that sent the LSP does not support partition repair.<br>• 1 – The IS that sent the LSP supports partition repair. |
| OL | The value in the LSP database overload field of the LSP. The field can have one of the following values:<br>• 0 – The overload bit is off.<br>• 1 – The overload bit is on, indicating that the IS that sent the LSP is overloaded and should not be used as a Level-2 router. |

To display detailed information for all the LSPs in the Foundry device's LSP databases, enter the following command at any level of the CLI:

```
BigIron MG8# show ipv6 isis database detail
IS-IS Level-1 Link State Database
LSPID                 LSP Seq Num  LSP Checksum  LSP Holdtime      ATT/P/OL
Router1.00-00         0x00000003   0x9a6b        566               0/0/0
   Area Address:  49.6561
   Metric:  10    IS Router2.01
   Metric:  12    IS Router1.02
   Metric:  10    IS Router1.03
   NLPID:  8e  cc
   IP address:  10.0.0.1
   Metric:  10    IP-Internal 110.10.0.0       255.255.0.0
   Metric:  10    IP-Internal 110.20.0.0       255.255.0.0
   Metric:  10    IP-Internal 110.30.0.0       255.255.0.0
    ...
     Hostname:  Router1
   IPv6 address:  3001::1
   Metric:  10    IPv6 Reachability 1111:5000::/32  UP bit: 0
   Metric:  10    IPv6 Reachability 1111:4000::/32  UP bit: 0
   Metric:  10    IPv6 Reachability 1111:3000::/32  UP bit: 0
...
```

**NOTE:** Portions of this display are truncated for brevity. The purpose of this display is to show all possible fields that might display rather than to show complete output.

***Syntax:*** show ipv6 isis database detail [l1 | l2 | level1 | level2]

The **l1** and **level1** parameters restrict the display to the Level-1 LSP entries. You can use these parameters interchangeably.

The **l2** and **level2** parameters restrict the display to the Level-2 LSP entries. You can use these parameters interchangeably.

For example, to display details about Level-1 LSPs only, enter a command such as the following at any CLI level:

```
BigIron MG8# show ipv6 isis database detail l1
```

This display shows the following information:

**Table 7.5: IPv6 IS-IS detailed LSP database information**

| This Field... | Displays... |
|---|---|
| LSPID | See the description in Table 7.4 on page 7-28. |
| LSP Seq Num | See the description in Table 7.4 on page 7-28. |
| LSP Checksum | See the description in Table 7.4 on page 7-28. |
| LSP Holdtime | See the description in Table 7.4 on page 7-28. |
| ATT/P/OL | See the description in Table 7.4 on page 7-28. |
| Area Address | The address of the area. |

**Table 7.5: IPv6 IS-IS detailed LSP database information(Continued)**

| This Field... | Displays... |
|---|---|
| TLVs | The remaining output displays the type, length, and value (TLV) parameters included in the LSPs. These parameters advertise reachability to IPv6 devices or networks. For example:<br>• A router identified as an IS and its hostname (Router2.01) can be reached using the default metric of 10.<br>• An end system within the current area identified as an IP-Internal and with the IP address of 110.10.0.0 and sub-net mask of 255.255.0.0 can be reached using the default metric of 10.<br>• An IPv6 prefix of 1111:5000::/32 is up and can be reached using the default metric of 10. |
| NLPID | The Network Layer Protocol Identifier (NLPID), which specifies the protocol the IS that sent the LSP is using. Usually, this value is “cc” but can also be “iso”. |
| IP address | The IP address of the interface that sent the LSP. The Foundry device can use this address as the next hop in routes to the addresses listed in the rows below. |
| Hostname | The hostname of the router that contains the LSP database that is displayed. |
| IPv6 address | The IPv6 address of the interface that sent the LSP. The Foundry device can use this address as the next hop in routes to the addresses listed in the rows below. |

## Displaying the System ID to Name Mappings

IS-IS maps the IS-IS system IDs to the hostnames of the devices with those IS. To display these mappings, enter the following command at any level of the CLI:

```
BigIron MG8# show ipv6 isis hostname
Total number of entries in IS-IS Hostname Table: 2
   System ID       Hostname            * = local IS
 * 2222.2222.2222  Router2
   1111.1111.1111  Router1
```

***Syntax:*** show ipv6 isis hostname

This example contains two mappings for this device. The Foundry device’s IS-IS system ID is “2222.2222.2222“and its hostname is “Router2”. The display contains an entry for another router. The display contains one entry for each IS that supports name mapping.

**NOTE:** Name mapping is enabled by default. When name mapping is enabled, the output of the **show ipv6 isis database** and **show ipv6 isis neighbor** commands uses the hostname instead of the system ID. To disable mapping so that these displays use the system ID instead, enter the **no hostname** command at the IS-IS router configuration level. For more information about performing this task, see the “Configuring IS-IS” chapter in the *Foundry NetIron Service Provider Configuration and Management Guide*.

## Displaying IPv6 IS-IS Interface Information

To display information about the interfaces on which IPv6 IS-IS is enabled, enter the following command at any level of the CLI:

```
BigIron MG8# show ipv6 isis interface
Total number of IS-IS Interfaces: 4

Interface : 2/1     Local Circuit Number: 00000001
   Circuit Type : BCAST Circuit Mode : LEVEL-1-2
   Circuit State: UP Passive State: FALSE
   MTU : 1497
   Level-1 Metric: 10, Level-1 Priority: 64
   Level-1 Hello Interval: 10 Level-1 Hello Multiplier: 3
   Level-1 Designated IS: Router2.01-22         Level-1 DIS Changes: 8
   Level-2 Metric: 10, Priority: 64
   Level-2 Hello Interval: 10 Level-2 Hello Multiplier: 3
   Level-2 Designated IS: Router2.01-00 Level-2 DIS Changes: 8
   Next IS-IS LAN Level-1 Hello in 1 seconds
   Next IS-IS LAN Level-2 Hello in 1 seconds
   Number of active Level-1 adjacencies: 1
   Number of active Level-2 adjacencies: 1
  Circuit State Changes: 0 Circuit Adjacencies State Changes: 2
   Rejected Adjacencies: 0
   Circuit Authentication Fails: 0 Bad LSP 0
   Control Messages Sent: 1696 Control Messages Received: 159
   IP Enabled: TRUE
   IP Address and Subnet Mask:
      10.0.0.2           255.0.0.0
     192.147.201.150     255.255.255.0
  IPv6 Enabled: TRUE
   IPv6 Address :
     3001::2
. . .
```

**NOTE:** The latter part of this display is truncated for brevity. The purpose of this display is to show all possible fields that might display rather than to show complete output.

***Syntax:*** show ipv6 isis interface

This display shows the following information:

**Table 7.6: IPv6 IS-IS interface information**

| This Field... | Displays... |
|---|---|
| Total number of IS-IS interfaces | The number of interfaces on which IPv6 IS-IS is enabled. |
| Interface | The port or virtual interface number to which the information listed below applies. |
| Local Circuit Number | The ID that the instance of IPv6 IS-IS running on the interface applied to the circuit between this interface and the interface at the other end of the link. |

**Table 7.6: IPv6 IS-IS interface information (Continued)**

| This Field... | Displays... |
|---|---|
| Circuit Type | The type of IS-IS circuit running on the interface. The circuit type can be one of the following:<br>• BCAST– broadcast<br>• PTP – point-to-point |
| Circuit Mode | The IS-IS type in use on the circuit. The mode can be one of the following:<br>• LEVEL-1<br>• LEVEL-2<br>• LEVEL-1-2 |
| Circuit State | The state of the circuit, which can be one of the following:<br>• DOWN<br>• UP |
| Passive State | The state of the passive option, which determines whether the interface is allowed to form an IS-IS adjacency with the IS at the other end of the circuit. The state can be one of the following:<br>• FALSE – The passive option is disabled. The interface can form an adjacency with the IS at the other end of the link.<br>• TRUE – The passive option is enabled. The interface cannot form an adjacency, but can still advertise itself into the area. |
| MTU | The maximum length supported for IS-IS PDUs sent on this interface. |
| Level-1 Metric | The default-metric value that the Foundry device inserts in IS-IS Level-1 PDUs originated on this interface. |
| Level-1 Priority | The priority of this IS to be elected as the Designated IS for Level-1 in this broadcast network. |
| Level-1 Hello Interval | The number of seconds the software waits between sending Level-1 hello PDUs to the IS at the other end of the circuit. |
| Level-1 Hello Multiplier | The number by which the software multiplies the hello interval to calculate the hold time for Level-1 Hello messages received on the circuit. |
| Level-1 Designated IS | The NET of the Level-1 Designated IS. |
| Level-1 DIS Changes | The number of times the NET of the Level-1 Designated IS has changed. |
| Level-2 Metric | The default-metric value that the router inserts in IS-IS Level-2 PDUs originated on this interface. |
| Level-2 Priority | The priority of this IS to be elected as the Designated IS for Level-2 in this broadcast network. |
| Level-2 Hello Interval | The number of seconds the software waits between sending Level-2 Hello messages to the IS at the other end of the circuit. |

**Table 7.6: IPv6 IS-IS interface information (Continued)**

| This Field... | Displays... |
|---|---|
| Level-2 Hello Multiplier | The number by which the software multiplies the hello interval to calculate the hold time for Level-2 LSPs received on the circuit. |
| Level-2 Designated IS | The NET of the Level-2 Designated IS. |
| Level-2 DIS Changes | The number of times the NET of the Level-2 Designated IS has changed. |
| Next IS-IS LAN Level-1 Hello | Number of seconds before next Level-1 Hello message will be transmitted by the Foundry device. |
| Next IS-IS LAN Level-2 Hello | Number of seconds before next Level-2 Hello message will be transmitted by the Foundry device. |
| Number of active Level-1 adjacencies | The number of ISs with which this interface has an active Level-1 adjacency. |
| Number of active Level-2 adjacencies | The number of ISs with which this interface has an active Level-2 adjacency. |
| Circuit State Changes | The number of times the state of the circuit has changed. |
| Circuit State Adjacencies Changes | The number of times an adjacency has started or ended on this circuit. |
| Rejected Adjacencies | The number of adjacency attempts by other ISs rejected by the router. |
| Circuit Authentication Fails | The number of times the Foundry device rejected a circuit because the authentication did not match the authentication configured on the Foundry device. |
| Bad LSP | The number of times the interface received a bad LSP from an IS at the other end of the circuit. The following conditions can cause an LSP to be bad:<br>• Invalid checksum<br>• Invalid length<br>• Invalid lifetime value |
| Control Messages Sent | The number of IS-IS control PDUs sent on this interface. |
| Control Messages Received | The number of IS-IS control PDUs received on this interface. |
| IP Enabled | The state of IP on the interface, which can be one of the following:<br>• TRUE – IP is enabled.<br>• FALSE – IP is disabled. |
| IP Address and Subnet Mask | The IP address(es) and sub-net mask(s) configured on this interface. |
| IPv6 Enabled | The state of IPv6 on the interface, which can be one of the following:<br>• TRUE – IPv6 is enabled.<br>• FALSE – IPv6 is disabled. |
| IPv6 Address | The IPv6 address(es) configured on this interface. |

## Displaying IPv6 IS-IS Memory Usage

To display information about IPv6 IS-IS memory usage, enter the following command at any level of the CLI:

```
BigIron MG8# show ipv6 isis memory
Total Static Memory Allocated : 1333 bytes
Total Dynamic Memory Allocated : 157952 bytes
Memory Type               Size       Allocated  Max-alloc  Alloc-Fails
MTYPE_ISIS_IP6_SUMMARY_PR 0          0          0          0
MTYPE_ISIS_OTHER          20         0          1          0
MTYPE_ISIS_IP6_ROUTE_NODE 21         22         1024       0
MTYPE_ISIS_IP6_ROUTE_INFO 12         17         1024       0
MTYPE_ISIS_IP6_NEXTHOP    24         2          256        0
MTYPE_ISIS_IP6_REDIS_ROUT 12         5          256        0
```

***Syntax:*** show ipv6 isis memory

This display shows the following information:

**Table 7.7: IPv6 IS-IS memory usage information**

| This Field... | Displays... |
|---|---|
| Total Static Memory Allocated | A summary of the amount of static memory allocated, in bytes, to IPv6 IS-IS. |
| Total Dynamic Memory Allocated | A summary of the amount of dynamic memory allocated, in bytes, to IPv6 IS-IS. |
| Memory Type | The type of memory used by IPv6 IS-IS. (This information is for use by Foundry's technical support in case of a problem.) |
| Size | The size of a memory type. |
| Allocated | The amount of memory currently allocated to a memory type. |
| Max-alloc | The maximum amount of memory that was allocated to a memory type. |
| Alloc-Fails | The number of times an attempt to allocate memory to a memory type failed. |

## Displaying IPv6 IS-IS Neighbor Information

You can display a summary or detailed information for all neighbors with which the Foundry device has formed an IS-IS adjacency.

To display a summary of all IPv6 IS-IS neighbors of a router, enter the following command at any level of the CLI:

```
BigIron MG8# show ipv6 isis neighbor
Total number of IS-IS Neighbors: 2
System Id      Interface   SNPA           State Holdtime Type Pri StateChgeTime
Router1        Ether 3/2   00e0.5200.0020 UP    30       ISL2 64  0  :0 :14:1
Router1        Ether 3/2   00e0.5200.0020 UP    30       ISL1 64  0  :0 :14:1
```

***Syntax:*** show ipv6 isis neighbor [detail]

This display shows the following information:

**Table 7.8: Summary of IPv6 IS-IS neighbor information**

| This Field... | Displays... |
|---|---|
| Total number of IS-IS Neighbors | The number of ISs with which the Foundry device has formed an IS-IS adjacency. |
| System ID | The system ID of the neighbor.<br>**Note:** Name mapping is enabled by default. When name mapping is enabled, the output of the **show ipv6 isis neighbor** command uses the hostname instead of the system ID. To disable mapping so that these displays use the system ID instead, enter the **no hostname** command at the IS-IS router configuration level. For more information about performing this task, see the "Configuring IS-IS" chapter in the *Foundry NetIron Service Provider Configuration and Management Guide*. |
| Interface | The router port or virtual interface attached to the neighbor. |
| SNPA | The Subnetwork Point of Attachment (SNPA), which is the MAC address of the Foundry device physical or virtual interface attached to the neighbor. |
| State | The state of the adjacency with the neighbor. The state can be one of the following:<br>• DOWN – The adjacency is down.<br>• INIT – The adjacency is being established and is not up yet.<br>• UP – The adjacency is up. |
| Holdtime | The time between transmissions of IS-IS hello messages. |
| Type | The IS-IS type of the adjacency. The type can be one of the following:<br>• ISL1 – Level-1 IS<br>• ISL2 – Level-2 IS<br>• PTP – Point-to-Point IS<br>• ES – ES<br>**Note**: The Foundry device forms a separate adjacency for each IS-IS type. Thus, if the router has both types of IS-IS adjacencies with the neighbor, the display contains a separate row of information for each adjacency. |
| Pri | The priority of this IS to be elected as the Designated IS in this broadcast network. |
| StateChgeTime | The amount of time that has passed since the adjacency last changed state. |

To display detailed information about all IPv6 IS-IS neighbors of a router, enter the following command at any level of the CLI:

```
BigIron MG8# show ipv6 isis neighbor detail
Total number of IS-IS Neighbors: 2
System ID      Interface   SNPA           State Holdtime Type Pri StateChgeTime
Router1        Ether 3/2   00e0.5200.0020 UP    30       ISL2 64  0  :0 :14:5
Area Address(es): 49.6561
IP Address(es): 10.0.0.1
IPv6 Address: fe80::2e0:52ff:fe00:20
Circuit ID: 2222.2222.2222.01
System ID      Interface   SNPA           State Holdtime Type Pri StateChgeTime
Router1        Ether 3/2   00e0.5200.0020 UP    30       ISL1 64  0  :0 :14:5
Area Address(es): 49.6561
IP Address(es): 10.0.0.1
IPv6 Address: fe80::2e0:52ff:fe00:20
Circuit ID: 2222.2222.2222.01
```

This display shows the following information:

**Table 7.9: Detailed IPv6 IS-IS neighbor information**

| This Field... | Displays... |
|---|---|
| Total number of IS-IS Neighbors | For information about this field, see Table 7.8 on page 7-35. |
| System ID | For information about this field, see Table 7.8 on page 7-35. |
| Interface | For information about this field, see Table 7.8 on page 7-35. |
| SNPA | For information about this field, see Table 7.8 on page 7-35. |
| State | For information about this field, see Table 7.8 on page 7-35. |
| Holdtime | For information about this field, see Table 7.8 on page 7-35. |
| Type | For information about this field, see Table 7.8 on page 7-35. |
| Pri | For information about this field, see Table 7.8 on page 7-35. |
| StateChgeTime | For information about this field, see Table 7.8 on page 7-35. |
| Area Address(es) | The address(es) of area(s) to which the neighbor interface belongs. |
| IP Address(es) | The IP address(es) assigned to the neighbor interface. |
| IPv6 Address | The IPv6 address(es) assigned to the neighbor interface. |
| Circuit ID | The ID of the IS-IS circuit running on the neighbor interface. |

## Displaying IPv6 IS-IS Path Information

To display information about all IPv6 IS-IS paths known to a router, enter the following command at any level of the CLI:

```
BigIron MG8# show ipv6 isis path-table
Prefix                                    Level      Metric
3001::/64                                 1          20
2222:5000::/32                            1          30
2222:4000::/32                            1          30
2222:3000::/32                            1          30
2222:2000::/32                            1          30
2222:1000::/32                            1          30
5555:1002::/32                            1          21
5555:2002::/32                            1          21
5555:3002::/32                            1          21
5555:4002::/32                            1          21
5555:5002::/32                            1          21
3002::/64                                 1          22
1111:5000::/32                            1          20
1111:4000::/32                            1          20
1111:3000::/32                            1          20
1111:2000::/32                            1          20
1111:1000::/32                            1          20
3001::/64                                 2          20
```

***Syntax:*** show ipv6 isis path-table

This display shows the following information:

**Table 7.10: IPv6 IS-IS path information**

| This Field... | Displays... |
|---|---|
| Prefix | The IPv6 paths known by the router. |
| Level | The level number associated with the path. Possible levels include the following:<br>• 1 – Level 1 path.<br>• 2 – Level 2 path.<br>• 1-2 – Level 1-2 path. |
| Metric | The value of the default metric, which is the IS-IS cost of using the area path to reach a destination. |

## Displaying IPv6 IS-IS Redistribution Information

To display information about the IPv6 routes redistributed into IPv6 IS-IS, enter the following command at any level of the CLI:

```
BigIron MG8# show ipv6 isis redistributed-routes
Prefix                                    Protocol  Level        Metric
5555:1002::/32                            Static    Level-2       1
5555:2002::/32                            Static    Level-2       1
5555:3002::/32                            Static    Level-2       1
5555:4002::/32                            Static    Level-2       1
5555:5002::/32                            Static    Level-2       1
```

***Syntax:*** show ipv6 isis redistributed-routes

This display shows the following information:

**Table 7.11: IPv6 IS-IS redistribution information**

| This Field... | Displays... |
|---|---|
| Prefix | The IPv6 routes redistributed into IPv6 IS-IS. |
| Protocol | The protocol from which the route is redistributed into IPv6 IS-IS. Possible protocols include the following:<br>• BGP – BGP4+.<br>• RIP – RIPng.<br>• OSPF – OSPFv3.<br>• Static – IPv6 static route table.<br>• Connected – A directly connected network. |
| Level | The IS-IS level into which a route is redistributed. Possible levels include the following:<br>• Level-1<br>• Level-2<br>• Level-1-2 |
| Metric | The value of the default redistribution metric, which is the IS-IS cost of redistributing the route into IPv6 IS-IS. |

## Displaying the IPv6 IS-IS Route Information

To display the routes in the router's IPv6 IS-IS route table, enter the following command at any level of the CLI:

```
BigIron MG8# show ipv6 isis routes
ISIS IPv6 Routing Table
Total Routes: 17  Level1: 17 Level2: 0 Equal-cost multi-path: 0
Type IPv6 Prefix                Next Hop Router           Interface  Cost
L1   1111:1000::/32             fe80::2e0:52ff:fe00:20    ethe 3/2   20
L1   1111:2000::/32             fe80::2e0:52ff:fe00:20    ethe 3/2   20
L1   1111:3000::/32             fe80::2e0:52ff:fe00:20    ethe 3/2   20
L1   1111:4000::/32             fe80::2e0:52ff:fe00:20    ethe 3/2   20
L1   1111:5000::/32             fe80::2e0:52ff:fe00:20    ethe 3/2   20
L1   2222:1000::/32             fe80::2e0:52ff:fe00:20    ethe 3/2   30
L1   2222:2000::/32             fe80::2e0:52ff:fe00:20    ethe 3/2   30
L1   2222:3000::/32             fe80::2e0:52ff:fe00:20    ethe 3/2   30
L1   2222:4000::/32             fe80::2e0:52ff:fe00:20    ethe 3/2   30
```

***Syntax:*** show ipv6 isis routes

This display shows the following information:

**Table 7.12: IPv6 IS-IS route information**

| This Field... | Displays... |
|---|---|
| Total Routes | The total number of routes in the router's IPv6 IS-IS route table. The total includes Level-1 and Level-2 routes. |
| Level1 | The total number of Level-1 routes in the IPv6 IS-IS route table. |
| Level2 | The total number of Level-1 routes in the IPv6 IS-IS route table. |
| Equal-cost multi-path | The total number of equal-cost routes to the same destination in the IPv6 IS-IS route table. If load sharing is enabled, the router equally distributes traffic among the routes. |
| Type | The route type, which can be one of the following:<br>• L1 – Level-1 route<br>• L2 – Level-2 route |
| IPv6 Prefix | The IPv6 prefix of the route. |
| Next Hop Router | The IPv6 address of the next-hop interface to the destination. |
| Interface | The router interface (physical or virtual interface) attached to the next hop. |
| Cost | The IPv6 IS-IS default metric for the route, which is the cost of using this route to reach the next-hop router to this destination. |

## Displaying IPv6 IS-IS Traffic Statistics

The router maintains statistics for common IS-IS PDU types. To display the IPv6 traffic statistics, enter the following command at any level of the CLI:

```
BigIron MG8# show ipv6 isis traffic
                          Message Received     Message Sent
Level-1 Hellos              98                   1171
Level-2 Hellos              96                   1170
PTP Hellos                  0                    0
Level-1 LSP                 3                    6
Level-2 LSP                 3                    6
Level-1 CSNP                1                    110
Level-2 CSNP                1                    110
Level-1 PSNP                0                    0
Level-2 PSNP                0                    0
```

***Syntax:*** show ipv6 isis traffic

This display shows the following information:

**Table 7.13: IPv6 IS-IS traffic statistics**

| This Field... | Displays... |
|---|---|
| Level-1 Hellos | The number of Level-1 hello PDUs sent and received by the router. |
| Level-2 Hellos | The number of Level-2 hello PDUs sent and received by the router. |
| PTP Hellos | The number of point-to-point hello PDUs sent and received by the router. |
| Level-1 LSP | The number of Level-1 link-state PDUs sent and received by the router. |
| Level-2 LSP | The number of Level-2 link-state PDUs sent and received by the router. |
| Level-1 CSNP | The number of Level-1 Complete Sequence Number PDUs (CSNPs) sent and received by the router. |
| Level-2 CSNP | The number of Level-2 CSNPs sent and received by the router. |
| Level-1 PSNP | The number of Level-1 Partial Sequence Number PDUs (PSNPs) sent and received by the router. |
| Level-2 PSNP | The number of Level-2 PSNPs sent and received by the router. |

# Chapter 8
# Configuring BGP4+

Foundry's implementation of IPv6 supports multi protocol BGP (MBGP) extensions, which allow IPv6 BGP (known as ***BGP4+***) to distribute routing information for protocols such as IPv4 BGP. The supported protocols are identified by address families. (For information about address families, see "Address Family Configuration Level" on page 8-1 and "Global and Address Family Configuration Levels" on page A-1.) The extensions allow a set of BGP4+ peers to exchange routing information for multiple address families and sub-address families.

IPv6 MBGP functions similarly to IPv4 MBGP except for the following enhancements:

- An IPv6 unicast address family and network layer reachability information (NLRI).
- Next hop attributes that use IPv6 addresses.

**NOTE:** Foundry's implementation of BGP4+ supports the advertising of routes among different address families. However, it supports BGP4+ unicast routes only; it does not currently support BGP4+ multicast routes.

This chapter describes the following:

- The address family configuration level for BGP4+.
- How to configure BGP4+.
- How to clear various BGP information, statistics, and counters.
- How to display BGP4+ information and statistics.

## Address Family Configuration Level

Foundry's implementation of BGP4+ includes a new configuration level: address family. For IPv6, Foundry currently supports the BGP4+ unicast address family configuration level only. (For IPv4, Foundry supports the BGP4 unicast and BGP4 multicast address family configuration levels.) The router enters the BGP4+ unicast address family configuration level when you enter the following command while at the global BGP configuration level:

```
BigIron(config-bgp)# address-family ipv6 unicast
BigIron(config-bgp-ipv6u)#
```

The (config-bgp-ipv6u)# prompt indicates that you are at the BGP4+ unicast address family configuration level.

While at the BGP4+ unicast address family configuration level, you can access several commands that allow you to configure BGP4+ unicast routes. The commands that you enter at this level apply only to IPv6 unicast address family only. You can generate a configuration for BGP4+ unicast routes that is separate and distinct from configurations for IPv4 unicast routes and IPv4 BGP multicast routes.

The commands that you can access while at the IPv6 unicast address family configuration level are also available at the IPv4 unicast and multicast address family configuration levels. Where relevant, this section discusses and provides IPv6-unicast-specific examples. For information about commands not discussed in this section, see the "Configuring BGP4" chapter in the *Foundry Enterprise Configuration and Management Guide*.

**NOTE:** Each address family configuration level allows you to access commands that apply to that particular address family only. To enable a feature in a particular address family, you must specify any associated commands for that feature in that particular address family. You cannot expect the feature, which you may have configured in the BGP4 unicast address family, to work in the BGP4+ unicast address family unless it is explicitly configured in the BGP4+ unicast address family.

To exit from the IPv6 unicast address family configuration level, enter the following command:

```
BigIron(config-bgp-ipv6u)# exit-address-family
BigIron(config-bgp)#
```

Entering this command returns you to the global BGP configuration level.

For complete information about the new CLI levels, see "Global and Address Family Configuration Levels" on page A-1.

# Configuring BGP4+

Before enabling BGP4+ on a router, you must enable the forwarding of IPv6 traffic on the router using the **ipv6 unicast-routing** command and enable IPv6 on at least one interface by configuring an IPv6 address or explicitly enabling IPv6 on that interface. For more information on performing these configuration tasks, see "Configuring Basic IPv6 Connectivity" on page 3-1.

To configure BGP4+, you must do the following:

- Enable BGP4+.
- Configure BGP4+ neighbors using one of the following methods:
  - Add one neighbor at a time (neighbor uses global or site-local IPv6 address).
  - Add one neighbor at a time (neighbor uses a link-local IPv6 address).
  - Create a peer group and add neighbors individually.

The following configuration tasks are optional:

- Advertise the default route.
- Import specified routes into BGP4+.
- Redistribute prefixes into BGP4+.
- Aggregate routes advertised to BGP4 neighbors.
- Use route maps.

## Enabling BGP4+

To enable BGP4+, enter commands such as the following:

```
BigIron(config)# router bgp
BGP: Please configure 'local-as' parameter in order to run BGP4.
BigIron(config-bgp)# local-as 1000
```

These commands enable the BGP4+ router and configure the autonomous system (1000) where the router resides.

***Syntax:*** [no] router bgp

To disable BGP, enter the **no** form of this command.

***Syntax:*** local-as <number>

Specify the AS number in which the router you are configuring resides.

After enabling BGP4+, you can add neighbors to a BGP4+ router by entering a commands such as the following:

```
BigIron(config-bgp)# address-family ipv6 unicast
BigIron(config-bgp-ipv6u)# neighbor 2001:4383:e0ff:783a::4 remote-as 1001
BigIron(config-bgp-ipv6u)# neighbor 2001:4383:e0ff:783a::5 remote-as 1001
```

These commands add two neighbors with global IPv6 addresses 2001:4383:e0ff:783a::4 and 2001:4383:e0ff:783a::5 in AS 1001.

---

**NOTE:** The example above adds IPv6 neighbors at the BGP4+ unicast address family configuration level. These neighbors, by default, are enabled to exchange BGP4+ unicast prefixes. However, if you add IPv6 neighbors while at the global BGP configuration/IPv4 BGP unicast address family configuration level, the neighbors will not exchange BGP4+ unicast prefixes until you explicitly enable them to do so by entering the **neighbor** <ipv6-address> | <peer-group-name> **activate** command at the BGP4+ unicast address family configuration level.

---

This section provides minimal information about adding BGP4+ neighbors, because its focus is to provide information about configuring BGP4+. For more information about the parameters you can use with this command, see the *Foundry Router Configuration Guide*.

## Configuring BGP4+ Neighbors Using Global or Site-Local IPv6 Addresses

To configure BGP4+ neighbors using global or site-local IPv6 addresses, you must add the IPv6 address of a neighbor in a remote AS to the BGP4+ neighbor table of the local router. You must repeat this procedure for each neighbor that you want to add to a local router.

For example, to add the IPv6 address 2011:f3e9:93e8:cc00::1 of a neighbor in remote AS 4500 to the BGP4+ neighbor table of a router, enter the following commands:

```
BigIron(config-bgp)# address-family ipv6 unicast
BigIron(config-bgp-ipv6u)# neighbor 2011:f3e9:93e8:cc00::1 remote-as 4500
```

***Syntax:*** neighbor <ipv6-address> remote-as <as-number>

---

**NOTE:** The example above adds an IPv6 neighbor at the BGP4+ unicast address family configuration level. This neighbor, by default, is enabled to exchange BGP4+ unicast prefixes. However, if you add an IPv6 neighbor while at the global BGP configuration/IPv4 BGP unicast address family configuration level, the neighbor will not exchange BGP4+ unicast prefixes until you explicitly enable it to do so by entering the **neighbor** <ipv6-address> | <peer-group-name> activate command at the BGP4+ unicast address family configuration level.

---

The **ipv6-address** parameter specifies the IPv6 address of the neighbor. You must specify the **ipv6-address** parameter in hexadecimal using 16-bit values between colons as documented in RFC 2373.

The **as-number** parameter indicates the number of the AS in which the neighbor resides.

To delete the neighbor from the BGP4+ neighbor table, enter the **no** form of this command.

## Adding BGP4+ Neighbors Using Link-Local Addresses

To configure BGP4+ neighbors that use link-local addresses, you must do the following:

- Add the IPv6 address of a neighbor in a remote AS to the BGP4+ neighbor table of the local router.
- Identify the neighbor interface over which the neighbor and local router will exchange prefixes.
- Configure a route map to set up a global next hop for packets destined for the neighbor.

### Adding BGP4+ Neighbor

To add the IPv6 link-local address fe80:4398:ab30:45de::1 of a neighbor in remote AS 1000 to the BGP4+ neighbor table of a router, enter the following commands:

```
BigIron(config-bgp)# address-family ipv6 unicast
BigIron(config-bgp-ipv6u)# neighbor fe80:4398:ab30:45de::1 remote-as 1000
```

***Syntax:*** neighbor <ipv6-address> remote-as <as-number>

**NOTE:** The example above adds an IPv6 neighbor at the BGP4+ unicast address family configuration level. This neighbor, by default, is enabled to exchange BGP4+ unicast prefixes. However, if you add an IPv6 neighbor while at the global BGP configuration/IPv4 BGP unicast address family configuration level, the neighbor will not exchange BGP4+ unicast prefixes until you explicitly enable it to do so by entering the **neighbor** <ipv6-address> | <peer-group-name> **activate** command at the BGP4+ unicast address family configuration level.

The <ipv6-address> parameter specifies the IPv6 link-local address of the neighbor. A link-local address has a fixed prefix of FE80::/10. You must specify the <ipv6-address> parameter in hexadecimal using 16-bit values between colons as documented in RFC 2373.

The <as-number> parameter indicates the number of the AS in which the neighbor resides.

To delete the neighbor from the BGP4+ neighbor table, enter the **no** form of this command.

### Identifying a Neighbor Interface

To specify Ethernet interface 3/1 as the neighbor interface over which the neighbor and local router will exchange prefixes, enter the following command:

```
BigIron(config-bgp)# neighbor fe80:4398:ab30:45de::1 update-source ethernet 3/1
```

***Syntax:*** neighbor <ipv6-address> update-source <ipv4-address> | ethernet <port> | loopback <number> | ve <number>

The <ipv6-address> parameter specifies the IPv6 link-local address of the neighbor. A link-local address has a fixed prefix of FE80::/10. You must specify the <ipv6-address> parameter in hexadecimal using 16-bit values between colons as documented in RFC 2373.

The <ipv4-address> parameter specifies the IPv4 address of the update source.

The **ethernet | loopback | ve** parameter specifies the neighbor interface over which the neighbor and local router will exchange prefixes. If you specify an Ethernet interface, also specify the port number associated with the interface. If you specify a loopback or VE interface, also specify the loopback or VE number.

### Configuring a Route Map

To configure a route map that filters routes advertised to a neighbor or sets up a global next hop for packets destined for the neighbor with the IPv6 link-local address fe80:4393:ab30:45de::1, enter commands such as the following (start at the BGP4+ unicast address family configuration level):

```
BigIron(config-bgp-ipv6u)# neighbor fe80:4398:ab30:45de::1 route-map out next-hop
BigIron(config-bgp-ipv6u)# exit
BigIron(config)# route-map next-hop permit 10
BigIron(config-route-map)# match ipv6 address prefix-list next-hop-ipv6
BigIron(config-route-map)# set ipv6 next-hop 2011:e0ff:3764::34
```

This route map applies to the BGP4+ unicast address family under which the **neighbor route-map** command is entered. This route map applies to the outgoing routes on the neighbor with the IPv6 link-local address fe80:4393:ab30:45de::1. If an outgoing route on the neighbor matches the route map, the route is distributed via the next hop router with the global IPv6 address 2011:e0ff:3764::34.

***Syntax:*** neighbor <ipv6-address> route-map [in | out] <name>

The <ipv6-address> parameter specifies the IPv6 link-local address of the neighbor. A link-local address has a fixed prefix of FE80::/10. You must specify the <ipv6-address> parameter in hexadecimal using 16-bit values between colons as documented in RFC 2373.

The **in** keyword applies the route map to incoming routes. The **out** keyword applies the route map to outgoing routes.

The <name> parameter specifies a route map name.

***Syntax:*** route-map <name> deny | permit <sequence-number>

The **name** parameter specifies a route map name.

The **deny** keyword denies the distribution of routes that match the route map. The **permit** keyword permits the distribution of routes that match the route map.

The <sequence-number> parameter specifies a sequence number for the route map statement.

***Syntax:*** match ipv6 address prefix-list <name>

The **match ipv6 address prefix-list** command distributes any routes that have a destination IPv6 address permitted by a prefix list.

The <name> parameter specifies an IPv6 prefix list name.

***Syntax:*** set ipv6 next-hop <ipv6-address>

The <ipv6-address> parameter specifies the IPv6 global address of the next-hop router. You must specify the <ipv6-address> parameter in hexadecimal using 16-bit values between colons as documented in RFC 2373.

## Configuring a BGP4+ Peer Group

If a router has multiple neighbors with similar attributes, you can configure a peer group, then add neighbors to the group instead of configuring neighbors individually for all parameters as described in “Configuring BGP4+ Neighbors Using Global or Site-Local IPv6 Addresses” on page 8-3 and “Adding BGP4+ Neighbors Using Link-Local Addresses” on page 8-3.

**NOTE:** You can add IPv6 neighbors only to an IPv6 peer group. You cannot add an IPv4 neighbor to an IPv6 peer group and vice versa. IPv6 and IPv6 peer groups must remain separate.

To configure a BGP4+ peer group, you must do the following:

1. Create a peer group.
2. Add a neighbor to the local router.
3. Assign the IPv6 neighbor to the peer group.

### Creating a BGP4+ Peer Group

To create a peer group named “peer_group1,” enter the following commands:

```
BigIron(config-bgp)# address-family ipv6 unicast
BigIron(config-bgp-ipv6u)# neighbor peer_group1 peer-group
```

***Syntax:*** neighbor <peer-group-name> peer-group

Specify a name for the peer group.

To delete the peer group, enter the **no** form of this command.

### Adding a Neighbor to a Local Router

To add the IPv6 address 2001:efff:89::23 of a neighbor in remote AS 1001 to the BGP4+ neighbor table of a router, enter the following command:

```
BigIron(config-bgp-ipv6u)# neighbor 2001:efff:89::23 remote-as 1001
```

**NOTE:** The example above adds an IPv6 neighbor at the BGP4+ unicast address family configuration level. This neighbor, by default, is enabled to exchange BGP4+ unicast prefixes. However, if you add an IPv6 neighbor while at the global BGP configuration/IPv4 BGP unicast address family configuration level, the neighbor will not exchange BGP4+ unicast prefixes until you explicitly enable it to do so by entering the **neighbor** <ipv6-address> | <peer-group-name> **activate** command at the BGP4+ unicast address family configuration level.

***Syntax:*** neighbor <ipv6-address> remote-as <as-number>

The **ipv6-address** parameter specifies the IPv6 address of the neighbor. You must specify the <ipv6-address> parameter in hexadecimal using 16-bit values between colons as documented in RFC 2373.

The <as-number> parameter indicates the number of the AS in which the neighbor resides.

To delete the neighbor from the BGP4+ neighbor table, enter the **no** form of this command.

### Assigning IPv6 Neighbor to Peer Group

To assign an already configured neighbor (2001:efff:89::23) to the peer group peer_group1, enter the following command at the BGP4+ unicast address family configuration level:

```
BigIron(config-bgp-ipv6u)# neighbor 2001:efff:89::23 peer-group peer_group1
```

***Syntax:*** neighbor <ipv6-address> peer-group <peer-group-name>

The <ipv6-address> parameter specifies the IPv6 address of the neighbor. You must specify the <ipv6-address> parameter in hexadecimal using 16-bit values between colons as documented in RFC 2373.

The **peer-group** <peer-group-name> parameter indicates the name of the already created peer group.

To delete the mapping of the neighbor IPv6 address to the peer group, enter the **no** form of this command.

## Advertising the Default BGP4+ Route

By default, the BGP4+ router does not originate and advertise a default BGP4+ route. A default route is the IPv6 address :: and the route prefix 0; that is, ::/0.

You can enable the BGP4+ router to advertise the default BGP4+ route by specifying the **default-information-originate** command at the BGP4+ unicast address family configuration level. Before entering this command, the default route ::/0 must be present in the IPv6 route table.

To enable the BGP4+ router to advertise the default route, enter the following command:

```
BigIron(config-bgp-ipv6u)# default-information-originate
```

***Syntax:*** [no] default-information-originate

You can also enable the BGP4+ router to send the default route to a particular neighbor by specifying the **neighbor** <ipv6-address> **default-originate** command at the BGP4+ unicast address family configuration level. This command does not require the presence of the default route ::/0 in the IPv6 route table.

For example, to enable the BGP4+ router to send the default route to a neighbor with the IPv6 address of 2001:efff:89::23, enter a command such as the following:

```
BigIron(config-bgp-ipv6u)# neighbor 2001:efff:89::23 default-originate
```

***Syntax:*** [no] neighbor <ipv6-address> default-originate [route-map <name>]

The <ipv6-address> parameter specifies a neighbor by its IPv6 address. You must specify this address in hexadecimal using 16-bit values between colons as documented in RFC 2373.

Specifying the optional **route-map** <name> parameter injects the default route conditionally, based on the match conditions in the route map.

## Importing Routes into BGP4+

By default, the router does not import routes into BGP4+. This section explains how to use the **network** command to enable the importing of specified routes into BGP4+.

**NOTE:** The routes imported into BGP4+ must first exist in the IPv6 unicast route table.

For example, to import the IPv6 prefix 3ff0:ec21::/32 into the BGP4+ database, enter the following command at the BGP4+ unicast address family configuration level:

```
BigIron(config-bgp-ipv6u)# network 3ff0:ec21::/32
```

***Syntax:*** network <ipv6-prefix>/<prefix-length> [route-map <name>]

You must specify the <ipv6-prefix> parameter in hexadecimal using 16-bit values between colons as documented in RFC 2373.

You must specify the <prefix-length> parameter as a decimal value. A slash mark (/) must follow the <ipv6-prefix> parameter and precede the <prefix-length> parameter.

You can specify the optional **route-map** <name> parameter if you want to change attributes of a route when importing it into BGP4+.

To disable the importing of a specified route, enter the **no** form of this command without the route-map parameter.

## Redistributing Prefixes into BGP4+

You can configure the router to redistribute routes from the following sources into BGP4+:

- Static IPv6 routes.
- Directly connected IPv6 networks.
- IPv6 IS-IS.
- OSPFv3.
- RIPng.

You can redistribute routes in the following ways:

- By route types, for example, the router redistributes all IPv6 static and RIPng routes.
- By using a route map to filter which routes to redistribute, for example, the router redistributes specified IPv6 static and RIPng routes only.

For example, to configure the redistribution of all RIPng routes into the BGP4+ unicast database, enter the following commands at the BGP4+ address family configuration level:

```
BigIron(config-bgp-ipv6u)# redistribute rip
```

***Syntax:*** redistribute <protocol> [level-1 | level-1-2 | level-2] [match external1 | external2 | internal] [metric <metric-value>] [route-map <name>]

The <protocol> parameter can be **connected**, **isis**, **ospf**, **rip**, or **static**.

If you specify **isis** as the protocol, you can optionally specify the redistribution of level 1, level 1 and 2, or level 2 routes.

If you specify **ospf** as the protocol, you can optionally specify the redistribution of external 1, external 2, or internal routes. (The default is internal.)

The **metric** <metric-value> parameter specifies the metric used for the redistributed route. If a value is not specified for this option, and no value is specified using the **default-metric** command at the BGP4+ unicast address family configuration level, the metric value for the IPv6 static, RIPng, or IPv6 OSPF route is used. Use a value consistent with the destination protocol. For more information about the **default-metric** command, see the "Configuring BGP4" chapter in the *Foundry Router Configuration Guide*.

The <name> parameter specifies a route map name.

## Aggregating Routes Advertised to BGP4 Neighbors

By default, a router advertises individual BGP4+ routes for all the networks. The aggregation feature allows you to configure a router to aggregate routes in a range of networks into a single IPv6 prefix. For example, without aggregation, a router will individually advertise routes for networks ff00:f000:0001:0000::/64, ff00:f000:0002:0000::/64,ff00:f000:0003:0000::/64, and so on. You can configure the router to instead send a single, aggregate route for the networks. The aggregate route would be advertised as ff00:f000::/24 to BGP4 neighbors.

To aggregate BGP4+ routes for ff00:f000:0001:0000::/64, ff00:f000:0002:0000::/64,ff00:f000:0003:0000::/64, enter the following command:

```
BigIron(config-bgp)# aggregate-address ff00:f000::/24 summary-only
```

***Syntax:*** aggregate-address <ipv6-prefix>/<prefix-length> [as-set] [summary-only] [suppress-map <map-name>] [advertise-map <map-name>] [attribute-map <map-name>]

The <ipv6-prefix>/<prefix-length> parameter specifies the aggregate value for the networks. You must specify the <ipv6-prefix> parameter in hexadecimal using 16-bit values between colons as documented in RFC 2373. You must specify the <prefix-length> parameter as a decimal value. A slash mark (/) must follow the <ipv6-prefix> parameter and precede the <prefix-length> parameter.

The **as-set** keyword causes the router to aggregate AS-path information for all the routes in the aggregate address into a single AS-path.

The **summary-only** keyword prevents the router from advertising more specific routes contained within the aggregate route.

The **suppress-map** <map-name> parameter prevents the more specific routes contained in the specified route map from being advertised.

The **advertise-map** <map-name> parameter configures the router to advertise the more specific routes in the specified route map.

The **attribute-map** <map-name> parameter configures the router to set attributes for the aggregate routes based on the specified route map.

---

**NOTE:** For the **suppress-map**, **advertise-map**, and **attribute-map** parameters, the route map must already be defined.

---

To remove an aggregate route from a BGP4 neighbor advertisement, use the **no** form of this command without any parameters.

### Using Route Maps

You can use a route map to filter and change values in BGP4+ routes. Currently, you can apply a route map to IPv6 unicast routes that are independent of IPv4 routes.

To configure a route map to match on IPv6 unicast routes, enter commands such as the following:

```
BigIron(config)# router bgp
BigIron(config-bgp)# address-family ipv6 unicast
BigIron(config-bgp-ipv6u)# neighbor 2001:eff3:df78::67 remote-as 1001
BigIron(config-bgp-ipv6u)# neighbor 2001:eff3:df78::67 route-map in map1
BigIron(config-bgp-ipv6u)# exit
BigIron(config)# ipv6 prefix-list ipv6_uni seq 10 permit 2001:eff3::/32
BigIron(config)# route-map map1 permit 10
BigIron(config-routemap-map1)# match ipv6 address prefix-list ipv6_uni
```

This example configures a route map named “map1” that permits incoming IPv6 unicast routes that match the prefix list named “ipv6_uni” (2001:eff3::/32). Note that you apply the route map while at the BGP4+ unicast address family configuration level.

## Clearing BGP4+ Information

This section contains information about clearing the following for BGP4+:

- Route flap dampening.
- Route flap dampening statistics.
- Neighbor information.
- BGP4+ routes in the IPv6 route table.
- Neighbor traffic counters.

**NOTE:** The **clear** commands implemented for BGP4+ correspond to the **clear** commands implemented for IPv4 BGP. For example, you can specify the **clear ipv6 bgp flap-statistics** command for IPv6 and the **clear ip bgp flap-statistics** for IPv4.

## Removing Route Flap Dampening

You can un-suppress routes by removing route flap dampening from the routes. The router allows you to un-suppress all routes at once or un-suppress individual routes.

To un-suppress all the suppressed routes, enter the following command at the Privileged EXEC level or any of the Config levels of the CLI:

```
BigIron# clear ipv6 bgp dampening
```

***Syntax:*** clear ipv6 bgp dampening [<ipv6-prefix>/<prefix-length>]

You must specify the <ipv6-prefix> parameter in hexadecimal using 16-bit values between colons as documented in RFC 2373.

You must specify the <prefix-length> parameter as a decimal value. A slash mark (/) must follow the <ipv6-prefix> parameter and precede the <prefix-length> parameter.

To un-suppress a specific route, enter a command such as the following:

```
BigIron# clear ipv6 bgp dampening 2001:e0ff::/32
```

This command un-suppresses only the route(s) for network 2001:e0ff::/32.

## Clearing Route Flap Dampening Statistics

The router allows you to clear all route flap dampening statistics or statistics for a specified IPv6 prefix or a regular expression.

**NOTE:** Clearing the dampening statistics for a route does not change the dampening status of the route.

To clear all the route dampening statistics, enter the following command at the Privileged EXEC level or any of the Config levels of the CLI:

```
BigIron# clear ipv6 bgp flap-statistics
```

***Syntax:*** clear ipv6 bgp flap-statistics [<ipv6-prefix>/<prefix-length> | neighbor <ipv6-address> | regular-expression <regular-expression>]

The <ipv6-prefix>/<prefix-length> parameter clears route flap dampening statistics for a specified IPv6 prefix. You must specify the <ipv6-prefix> parameter in hexadecimal using 16-bit values between colons as documented in RFC 2373. You must specify the <prefix-length> parameter as a decimal value. A slash mark (/) must follow the <ipv6-prefix> parameter and precede the <prefix-length> parameter.

The **neighbor** <ipv6-address> parameter clears route flap dampening statistics only for routes learned from the neighbor with the specified IPv6 address.

The **regular-expression** <regular-expression> parameter is a regular expression. For more information about regular expressions, see the “Configuring BGP4” chapter in the *Foundry Router Configuration Guide*.

## Clearing BGP4+ Local Route Information

You can clear locally imported or routes redistributed into BGP4+.

To clear all local route information, enter the following command at the Privileged EXEC level or any of the Config levels of the CLI:

```
BigIron# clear ipv6 bgp local routes
```

***Syntax:*** clear ipv6 bgp local routes

## Clearing BGP4+ Neighbor Information

You can perform the following tasks related to BGP4+ neighbor information:

- Clear diagnostic buffers.
- Reset a session to send and receive Outbound Route Filters (ORFs).
- Close a session, or reset a session and resend/receive an update.
- Clear traffic counters.
- Clear route flap dampening statistics.

### Clearing BGP4+ Neighbor Diagnostic Buffers

You can clear the following BGP4+ neighbor diagnostic information in buffers:

- The first 400 bytes of the last packet that contained an error.
- The last NOTIFICATION message either sent or received by the neighbor.

To display these buffers, use the **last-packet-with-error** keyword with the **show ipv6 bgp neighbors** command. For more information about this command, see “Displaying Last Error Packet from a BGP4+ Neighbor” on page 8-43.

You can clear the buffers for all neighbors, for an individual neighbor, or for all the neighbors within a specific peer group or AS.

To clear these buffers for neighbor 2000:e0ff:37::1, enter the following commands at the Privileged EXEC level or any of the Config levels of the CLI:

```
BigIron# clear ipv6 bgp neighbor 2000:e0ff:37::1 last-packet-with-error
BigIron# clear ipv6 bgp neighbor 2000:e0ff:37::1 notification-errors
```

***Syntax:*** clear ipv6 bgp neighbor all | <ipv6-address> | <peer-group-name> | <as-number> last-packet-with-error | notification-errors

The **all** | <ipv6-address> | <peer-group-name> | <as-num> specifies the neighbor. The <ipv6-address> parameter specifies a neighbor by its IPv6 address. You must specify this address in hexadecimal using 16-bit values between colons as documented in RFC 2373. The <peer-group-name> specifies all neighbors in a specific peer group. The <as-num> parameter specifies all neighbors within the specified AS. The **all** keyword specifies all neighbors.

The **last-packet-with-error** keyword clears the buffer containing the first 400 bytes of the last packet that contained errors.

The **notification-errors** keyword clears the notification error code for the last NOTIFICATION message sent or received.

### Resetting a BGP4+ Neighbor Session to Send and Receive ORFs

You can perform a hard or soft reset of a BGP4+ neighbor session to send or receive ORFs. For more information about cooperative filtering, see the “Configuring BGP4” chapter in the *Foundry Router Configuration Guide*.

To perform a hard reset of a neighbor session and send ORFs to the neighbor, enter a command such as the following:

```
BigIron# clear ipv6 bgp neighbor 2000:e0ff:38::1
```

This command resets the BGP4+ session with neighbor 2000:e0ff:38::1 and sends the ORFs to the neighbor when the neighbor comes up again. If the neighbor sends ORFs to the router, the router accepts them if the send capability is enabled.

To perform a soft reset of a neighbor session and send ORFs to the neighbor, enter a command such as the following:

```
BigIron(config)# clear ipv6 bgp neighbor peer_group1 soft in prefix-list
```

***Syntax:*** clear ipv6 bgp neighbor <ipv6-address> | <peer-group-name> [soft in prefix-filter]

The <ipv6-address> parameter specifies a neighbor by its IPv6 address. You must specify this address in hexadecimal using 16-bit values between colons as documented in RFC 2373.

The <peer-group-name> specifies all neighbors in a specific peer group.

If you use the **soft in prefix-filter** keyword, the router sends an updated IPv6 prefix list to the neighbor as part of its route refresh message to the neighbor.

### Closing or Resetting a BGP4+ Neighbor Session

You can close a neighbor session or resend route updates to a neighbor. You can specify all neighbors, a single neighbor, or all neighbors within a specific peer group or AS.

If you close a neighbor session, the router and the neighbor clear all the routes they learned from each other. When the router and neighbor establish a new BGP4+ session, they exchange route tables again. Use this method if you want the router to relearn routes from the neighbor and resend its own route table to the neighbor.

If you use the **soft-outbound** keyword, the router compiles a list of all the routes it would normally send to the neighbor at the beginning of a session. However, before sending the updates, the router also applies the filters and route maps you have configured to the list of routes. If the filters or route maps result in changes to the list of routes, the router sends updates to advertise, change, or even withdraw routes on the neighbor as needed. This ensures that the neighbor receives only the routes you want it to contain. Even if the neighbor already contains a route learned from the router that you later decided to filter out, using the soft-outbound option removes that route from the neighbor. If no change is detected from the previously sent routes, an update is not sent.

For example, to close all neighbor sessions and thus flush all the routes exchanged by the router and all neighbors, enter the following command at the Privileged EXEC level or any of the Config levels of the CLI:

```
BigIron# clear ipv6 bgp neighbor all
```

***Syntax:*** clear ipv6 bgp neighbor all | <ipv6-address> | <peer-group-name> | <as-number> [soft-outbound | soft [in | out]]

The **all** | <ipv6-address> | <peer-group-name> | <as-number> specifies the neighbor. The <ipv6-address> parameter specifies a neighbor by its IPv6 address. You must specify this address in hexadecimal using 16-bit values between colons as documented in RFC 2373. The <peer-group-name> specifies all neighbors in a specific peer group. The <as-number> parameter specifies all neighbors within the specified AS. The **all** keyword specifies all neighbors.

Use the **soft-outbound** keyword to perform a soft reset of a neighbor session and resend only route update changes to a neighbor.

Use the **soft in** parameter to perform a soft reset of a neighbor session and requests a route update from a neighbor.

Use the **soft out** parameter to perform a soft reset of a neighbor session and resend all routes to a neighbor.

### Clearing BGP4+ Neighbor Traffic Counters

You can clear the BGP4+ message counter (reset them to 0) for all neighbors, a single neighbor, or all neighbors within a specific peer group or AS.

For example, to clear the BGP4+ message counter for all neighbors within an AS 1001, enter a command such as the following at the Privileged EXEC level or any of the Config levels of the CLI:

```
BigIron# clear ipv6 bgp neighbor 1001 traffic
```

***Syntax:*** clear ipv6 bgp neighbor all | <ipv6-address> | <peer-group-name> | <as-number> traffic

The **all** | <ipv6-address> | <peer-group-name> | <as-number> specifies the neighbor. The <ipv6-address> parameter specifies a neighbor by its IPv6 address. You must specify this address in hexadecimal using 16-bit values between colons as documented in RFC 2373. The <peer-group-name> specifies all neighbors in a specific peer group. The <as-number> parameter specifies all neighbors within the specified AS. The **all** keyword specifies all neighbors.

Specify the **traffic** keyword to clear the BGP4+ message counter.

### Clearing BGP4+ Neighbor Route Flap Dampening Statistics

The router allows you to clear all route flap dampening statistics for a specified BGP4+ neighbor.

**NOTE:** Clearing the dampening statistics for a neighbor does not change the dampening status of a route.

To clear all of the route flap dampening statistics for a neighbor, enter a command such as the following at the Privileged EXEC level or any of the Config levels of the CLI:

```
BigIron# clear ipv6 bgp neighbor 2000:e0ff:47::1 flap-statistics
```

***Syntax:*** clear ipv6 bgp neighbor <ipv6-address> flap-statistics

The <ipv6-address> parameter specifies a neighbor by its IPv6 address. You must specify this address in hexadecimal using 16-bit values between colons as documented in RFC 2373.

Specify the **flap-statistics** keyword to clear route flap dampening statistics for the specified neighbor.

## Clearing and Resetting BGP4+ Routes in the IPv6 Route Table

You can clear all BGP4+ routes or only those routes associated with a particular IPv6 prefix from the IPv6 route table and reset the routes. When cleared, the BGP4+ routes are removed from the IPv6 main route table and then restored again.

For example, to clear all BGP4+ routes and reset them, enter the following command at the Privileged EXEC level or any of the Config levels of the CLI:

```
BigIron# clear ipv6 bgp routes
```

***Syntax:*** clear ip bgp routes [<ipv6-prefix>/<prefix-length>]

The <ipv6-prefix>/<prefix-length> parameter clears routes associated with a particular IPv6 prefix. You must specify the <ipv6-prefix> parameter in hexadecimal using 16-bit values between colons as documented in RFC 2373. You must specify the <prefix-length> parameter as a decimal value. A slash mark (/) must follow the <ipv6-prefix> parameter and precede the <prefix-length> parameter.

## Clearing Traffic Counters for All BGP4+ Neighbors

To clear the message counters (reset them to 0) for all BGP4+ neighbors, enter the following command:

```
BigIron(config)# clear ipv6 bgp traffic
```

***Syntax:*** clear ipv6 bgp traffic

# Displaying BGP4+ Information

You can display the following BGP4+ information:

- BGP4+ route table.
- BGP4+ route information.
- BGP4+ route-attribute entries.
- BGP4+ configuration information.
- Dampened BGP4+ paths.
- Filtered-out BGP4+ routes.
- BGP4+ route flap dampening statistics.
- BGP4+ neighbor information.
- BGP4+ peer group configuration information.
- BGP4+ summary information.

**NOTE:** The **show** commands implemented for BGP4+ correspond to the **show** commands implemented for IPv4 BGP. For example, you can specify the **show ipv6 bgp** command for IPv6 and the **show ip bgp** command for IPv4. Also, the displays for the IPv4 and IPv6 versions of the **show** commands are similar except where relevant, IPv6 neighbor addresses replace IPv4 neighbor addresses, IPv6 prefixes replace IPv4 prefixes, and IPv6 next-hop addresses replace IPv4 next-hop addresses.

## Displaying the BGP4+ Route Table

BGP4+ uses filters you define, as well as an algorithm to determine the preferred route to a destination. (For information about the algorithm, see the "Configuring BGP4" chapter in *Foundry Enterprise Configuration and Management Guide*.) BGP4+ sends only the preferred route to the router's IPv6 table. However, if you want to view all the routes BGP4+ knows about, you can display the BGP4+ table.

To display the BGP4+ route table, enter the following command at any level of the CLI:

```
BigIron# show ipv6 bgp routes
Total number of BGP Routes: 2
Status A:AGGREGATE B:BEST b:NOT-INSTALLED-BEST C:CONFED_EBGP D:DAMPED
       E:EBGP H:HISTORY I:IBGP L:LOCAL M:MULTIPATH S:SUPPRESSED
       Prefix              Next Hop        Metric     LocPrf     Weight Status
1      2002::/16           ::              1          100        32768  BL
         AS_PATH:
2      2002:1234::/32      ::              1          100        32768  BL
         AS_PATH:
```

This display shows the following information:

**Table 8.1: Summary of BGP4+ routes**

| This Field... | Displays... |
|---|---|
| Number of BGP4+ Routes | The number of routes displayed by the command. |
| Status codes | A list of the characters the display uses to indicate the route's status. The status code appears in the Status column of the display. The status codes are described in the command's output. |
| Prefix | The route's prefix. |
| Next Hop | The next-hop router for reaching the route from the router. |
| Metric | The value of the route's MED attribute. If the route does not have a metric, this field is blank. |
| LocPrf | The degree of preference for the advertised route relative to other routes in the local AS. When the BGP4+ algorithm compares routes on the basis of local preferences, the route with the higher local preference is chosen. The preference can have a value from 0 – 4294967295. |
| Weight | The value that this router associates with routes from a specific neighbor. For example, if the router receives routes to the same destination from two BGP4+ neighbors, the router prefers the route from the neighbor with the larger weight. |

**Table 8.1: Summary of BGP4+ routes (Continued)**

| This Field... | Displays... |
|---|---|
| Status | The route's status, which can be one or more of the following:<br>• A – AGGREGATE. The route is an aggregate route for multiple networks.<br>• B – BEST. BGP4+ has determined that this is the optimal route to the destination.<br>• b – NOT-INSTALLED-BEST – BGP4+ has determined that this is the optimal route to the destination but did not install it in the IPv6 route table because the router received better routes from other sources (such as OSPFv3, RIPng, or static IPv6 routes).<br>• C – CONFED_EBGP. The route was learned from a neighbor in the same confederation and AS, but in a different sub-AS within the confederation.<br>• D – DAMPED. This route has been dampened (by the route dampening feature), and is currently unusable.<br>• E – EBGP. The route was learned through a router in another AS.<br>• H – HISTORY. Route dampening is configured for this route, and the route has a history of flapping and is unreachable now.<br>• I – IBGP. The route was learned through a router in the same AS.<br>• L – LOCAL. The route originated on this router.<br>• M – MULTIPATH. BGP4+ load sharing is enabled and this route was selected as one of the best ones to the destination. The best route among the multiple paths also is marked with "B".<br>**Note**: If the "m" is shown in lowercase, the software was not able to install the route in the IPv6 route table.<br>• S – SUPPRESSED. This route was suppressed during aggregation and thus is not advertised to neighbors. |
| AS-PATH | The AS-path information for the route. |

***Syntax:*** show ipv6 bgp routes [<ipv6-prefix>/<prefix-length> | <table-entry-number> | age <seconds> | as-path-access-list <name> | as-path-filter <number> | best | cidr-only | [community <number> | no-export | no-advertise | internet | local-as] | community-access-list <name> | community-filter <number> | detail [<option>] | local | neighbor <ipv6-address> | nexthop <ipv6-address> | no-best | prefix-list <name> | regular-expression <regular-expression> | route-map <name> | summary | unreachable]

You can use the following options with the **show ipv6 bgp routes** command to determine the content of the display:

The <ipv6-prefix>/<prefix-length> parameter displays routes for a specific network. You must specify the <ipv6-prefix> parameter in hexadecimal using 16-bit values between colons as documented in RFC 2373. You must specify the <prefix-length> parameter as a decimal value. A slash mark (/) must follow the <ipv6-prefix> parameter and precede the <prefix-length> parameter.

The <table-entry-number> parameter specifies the table entry with which you want the display to start. For example, if you specify 100, the display shows entry 100 and all entries subsequent to entry 100.

The **age** <seconds> parameter displays only the routes that have been received or updated more recently than the number of seconds you specify.

The **as-path-access-list** <name> parameter filters the display using the specified AS-path ACL.

The **as-path-filter** <number> parameter filters the display using the specified AS-path filter.

The **best** keyword displays the routes received from neighbors that the router selected as the best routes to their destinations.

The **cidr-only** keyword lists only the routes whose network masks do not match their class network length.

The **community** <number> parameter lets you display routes for a specific community. You can specify **local-as**, **no-export**, **no-advertise**, **internet**, or a private community number. You can specify the community number as either two five-digit integer values of up to 1– 65535, separated by a colon (for example, 12345:6789) or a single long integer value.

The **community-access-list** <name> parameter filters the display using the specified community ACL.

The **community-filter** <number> parameter lets you display routes that match a specific community filter.

The **detail** <option> parameter lets you display more details about the routes. You can refine your request by also specifying one of the other parameters after the **detail** keyword.

The **local** keyword displays routes that are local to the router.

The **neighbor** <ipv6-address> parameter displays routes learned from a specified BGP4+ neighbor.

The **nexthop** <ipv6-address> parameter displays the routes for a specified next-hop IPv6 address. You must specify this address in hexadecimal using 16-bit values between colons as documented in RFC 2373.

The **no-best** keyword displays the routes for which none of the routes to a given prefix were selected as the best route. The IPv6 route table does not contain a BGP4+ route for any of the routes listed using this option.

The **prefix-list** <name> parameter filters the display using the specified IPv6 prefix list.

The **regular-expression** <regular-expression> parameter filters the display based on a regular expression. For more information about regular expressions, see the “Configuring BGP4” chapter in the *Foundry Router Configuration Guide*.

The **route-map** <name> parameter filters the display using the specified route map. The software displays only the routes that match the match statements in the route map. The software disregards the route map’s set statements.

The **summary** keyword displays summary information for the routes.

The **unreachable** keyword displays the routes that are unreachable because the router does not have a valid RIPng, OSPFv3, IPv6 IS-IS, or static IPv6 route to the next hop.

To display details about BGP4+ routes, enter the following command at any level of the CLI:

```
BigIron# show ipv6 bgp routes detail
Total number of BGP Routes: 2
Status A:AGGREGATE B:BEST b:NOT-INSTALLED-BEST C:CONFED_EBGP D:DAMPED
       E:EBGP H:HISTORY I:IBGP L:LOCAL M:MULTIPATH S:SUPPRESSED
1       Prefix: 2002::/16,  Status: BL,  Age: 2d17h10m42s
         NEXT_HOP: ::,  Learned from Peer: Local Router
          LOCAL_PREF: 100,  MED: 1,  ORIGIN: incomplete,  Weight: 32768
           AS_PATH:
            Adj_RIB_out count: 1,  Admin distance 190
2       Prefix: 2002:1234::/32,  Status: BL,  Age: 2d17h10m42s
         NEXT_HOP: ::,  Learned from Peer: Local Router
          LOCAL_PREF: 100,  MED: 1,  ORIGIN: incomplete,  Weight: 32768
           AS_PATH:
            Adj_RIB_out count: 1,  Admin distance 190
```

This display shows the following information:

**Table 8.2: Detailed BGP4+ route information**

| This Field... | Displays... |
|---|---|
| Number of BGP4+ Routes advertised to specified neighbor (appears only in display for all routes) | For information about this field, see Table 8.1 on page 8-13. |
| Status codes | For information about this field, see Table 8.1 on page 8-13. |
| Prefix | For information about this field, see Table 8.1 on page 8-13. |
| Status | For information about this field, see Table 8.1 on page 8-13. |
| Age | The age of the advertised route, in seconds. |
| Next Hop | For information about this field, see Table 8.1 on page 8-13. |
| Learned from Peer | The IPv6 address of the neighbor from which this route is learned. “Local Router” indicates that the router itself learned the route. |
| LOCAL_PREF | For information about this field, see Table 8.1 on page 8-13. |
| MED | The value of the advertised route’s MED attribute. If the route does not have a metric, this field is blank. |

**Table 8.2: Detailed BGP4+ route information (Continued)**

| This Field... | Displays... |
|---|---|
| Origin | The source of the route information. The origin can be one of the following:<br>• A – AGGREGATE. The route is an aggregate route for multiple networks.<br>• B – BEST. BGP4+ has determined that this is the optimal route to the destination.<br>• b – NOT-INSTALLED-BEST – BGP4+ has determined that this is the optimal route to the destination but did not install it in the IPv6 route table because the router received better routes from other sources (such as OSPFv3, RIPng, or static IPv6 routes).<br>• C – CONFED_EBGP. The route was learned from a neighbor in the same confederation and AS, but in a different sub-AS within the confederation.<br>• D – DAMPED. This route has been dampened (by the route dampening feature), and is currently unusable.<br>• EGP – The routes with this set of attributes came to BGP4+ through EGP.<br>• H – HISTORY. Route dampening is configured for this route, and the route has a history of flapping and is unreachable now.<br>• IGP – The routes with this set of attributes came to BGP4+ through IGP.<br>• L – LOCAL. The route originated on this router.<br>• M – MULTIPATH. BGP4+ load sharing is enabled and this route was selected as one of the best ones to the destination. The best route among the multiple paths also is marked with “B”.<br>**Note**: If the “m” is shown in lowercase, the software was not able to install the route in the IPv6 route table.<br>• S – SUPPRESSED. This route was suppressed during aggregation and thus is not advertised to neighbors. |
| Weight | For information about this field, see Table 8.1 on page 8-13. |
| AS-PATH | For information about this field, see Table 8.1 on page 8-13. |
| Adj_RIB_out count | The number of neighbors to which the route has been or will be advertised. This is the number of times the route has been selected as the best route and placed in the Adj-RIB-Out (outbound queue) for a BGP4+ neighbor. |
| Admin Distance | The administrative distance of the route. |

***Syntax:*** show ipv6 bgp routes detail [<ipv6-prefix>/<prefix-length> | <table-entry-number> | age <seconds> | as-path-access-list <name> | as-path-filter <number> | best | cidr-only | [community <number> | no-export | no-advertise | internet | local-as] | community-access-list <name> | community-filter <number> | local | neighbor <ipv6-address> | nexthop <ipv6-address> | no-best | prefix-list <name> | regular-expression <regular-expression> | route-map <name> | summary | unreachable]

You can use the following options with the **show ipv6 bgp routes detail** command to determine the content of the display:

The <ipv6-prefix>/<prefix-length> option displays details about routes for a specific network. You must specify the <ipv6-prefix> parameter in hexadecimal using 16-bit values between colons as documented in RFC 2373. You must specify the <prefix-length> parameter as a decimal value. A slash mark (/) must follow the <ipv6-prefix> parameter and precede the <prefix-length> parameter.

The <table-entry-number> parameter specifies the table entry with which you want the display to start. For example, if you specify 100, the display shows entry 100 and all entries subsequent to entry 100.

The **age** <seconds> parameter displays only the routes that have been received or updated more recently than the number of seconds you specify.

The **as-path-access-list** <name> parameter filters the display using the specified AS-path ACL.

The **as-path-filter** <number> parameter filters the display using the specified AS-path filter.

The **best** keyword displays the routes received from neighbors that the router selected as the best routes to their destinations.

The **cidr-only** keyword lists only the routes whose network masks do not match their class network length.

The **community** <number> parameter lets you display routes for a specific community. You can specify **local-as**, **no-export**, **no-advertise**, **internet**, or a private community number. You can specify the community number as either two five-digit integer values of up to 1– 65535, separated by a colon (for example, 12345:6789) or a single long integer value.

The **community-access-list** <name> parameter filters the display using the specified community ACL.

The **community-filter** <number> parameter lets you display routes that match a specific community filter.

The **detail** keyword lets you display more details about the routes. You can refine your request by also specifying one of the other parameters after the **detail** keyword.

The **local** keyword displays routes that are local to the router.

The **neighbor** <ipv6-address> parameter displays routes learned from a specified BGP4+ neighbor.

The **nexthop** <ipv6-address> option displays the routes for a specified next-hop IPv6 address. You must specify this address in hexadecimal using 16-bit values between colons as documented in RFC 2373.

The **no-best** keyword displays the routes for which none of the routes to a given prefix were selected as the best route. The IPv6 route table does not contain a BGP4+ route for any of the routes listed using this option.

The **prefix-list** <name> parameter filters the display using the specified IPv6 prefix list.

The **regular-expression** <regular-expression> parameter filters the display based on a regular expression. For more information about regular expressions, see the “Configuring BGP4” chapter in the *Foundry Router Configuration Guide*.

The **route-map** <name> parameter filters the display using the specified route map. The software displays only the routes that match the match statements in the route map. The software disregards the route map’s set statements.

The **summary** keyword displays summary information for the routes.

The **unreachable** keyword displays the routes that are unreachable because the router does not have a valid RIPng, OSPFv3, IPv6 IS-IS, or static IPv6 route to the next hop.

## Displaying BGP4+ Route Information

You can display all BGP4+ routes known by a router, only those routes that match a specified prefix, or routes that match a specified or longer prefix.

To display all BGP4+ routes known by the router, enter the following command at any level of the CLI:

```
BigIron# show ipv6 bgp
Total number of BGP Routes: 2
Status codes: s suppressed, d damped, h history, * valid, > best, i internal
Origin codes: i - IGP, e - EGP, ? - incomplete
    Network            Next Hop        Metric LocPrf Weight Path
*>  2002::/16          ::              1      100    32768  ?
*>  2002:1234::/32     ::              1      100    32768  ?
```

***Syntax:*** show ipv6 bgp <ipv6-prefix>/<prefix-length> [longer-prefixes]

The <ipv6-prefix>/<prefix-length> parameter allows you to display routes that match a specified BGP prefix only. You must specify the <ipv6-prefix> parameter in hexadecimal using 16-bit values between colons as documented in RFC 2373. You must specify the <prefix-length> parameter as a decimal value. A slash mark (/) must follow the <ipv6-prefix> parameter and precede the <prefix-length> parameter.

The **longer-prefixes** keyword allows you to display routes that match a specified or longer BGP prefix. For example, if you specify **2002::/16 longer-prefixes**, then all routes with the prefix 2002::/16 or that have a longer prefix (such as 2002:e016::/32) are displayed.

To display only those routes that match prefix 2002::/16, enter the following command at any level of the CLI:

```
BigIron# show ipv6 bgp 2002::/16
Number of BGP Routes matching display condition : 1
Status codes: s suppressed, d damped, h history, * valid, > best, i internal
Origin codes: i - IGP, e - EGP, ? - incomplete
    Network            Next Hop        Metric LocPrf Weight Path
*>  2002::/16          ::              1      100    32768  ?
       Route is advertised to 1 peers:
        2000:4::110(65002)
```

For example, to display routes that match prefix 2002::/16 or longer, enter the following command at any level of the CLI:

```
BigIron# show ipv6 bgp 2002::/16 longer-prefixes
Number of BGP Routes matching display condition : 3
Status codes: s suppressed, d damped, h history, * valid, > best, i internal
Origin codes: i - IGP, e - EGP, ? - incomplete
    Network            Next Hop        Metric LocPrf Weight Path
*>  2002::/16          ::              1      100    32768  ?
*>  2002:1234::/32     ::              1      100    32768  ?
*>  2002:e0ff::/32     ::              1      100    32768  ?
       Route is advertised to 1 peers:
        2000:4::110(65002)
```

These displays show the following information:

**Table 8.3: BGP4+ route information**

| This Field... | Displays... |
|---|---|
| Total number of BGP Routes (appears in display of all BGP routes only) | The number of routes known by the router. |

**Table 8.3: BGP4+ route information (Continued)**

| This Field... | Displays... |
|---|---|
| Number of BGP Routes matching display condition (appears in display that matches specified and longer prefixes) | The number of routes that matched the display parameters you entered. This is the number of routes displayed by the command. |
| Status codes | A list of the characters the display uses to indicate the route's status. The status code appears in the left column of the display, to the left of each route. The status codes are described in the command's output. |
| Origin codes | A character the display uses to indicate the route's origin. The origin code appears to the right of the AS path (Path field). The origin codes are described in the command's output. |
| Network | The network prefix and prefix length. |
| Next Hop | The next-hop router for reaching the network from the router. |
| Metric | The value of the route's MED attribute. If the route does not have a metric, this field is blank. |
| LocPrf | The degree of preference for this route relative to other routes in the local AS. When the BGP4+ algorithm compares routes on the basis of local preferences, the route with the higher local preference is chosen. The preference can have a value from 0 – 4294967295. |
| Weight | The value that this router associates with routes from a specific neighbor. For example, if the router receives routes to the same destination from two BGP4+ neighbors, the router prefers the route from the neighbor with the larger weight. |
| Path | The route's AS path. |

## Displaying BGP4+ Route-Attribute Entries

The route-attribute entries table lists sets of BGP4+ attributes stored in the router's memory. Each set of attributes is unique and can be associated with one or more routes. In fact, the router typically has fewer route attribute entries than routes.

To display the IPv6 route-attribute entries table, enter the following command:

```
BigIron# show ipv6 bgp attribute-entries
        Total number of BGP Attribute Entries: 378
1       Next Hop  :::                Metric   :1              Origin:INCOMP
        Originator:0.0.0.0           Cluster List:None
        Aggregator:AS Number :0      Router-ID:0.0.0.0       Atomic:None
        Local Pref:100               Communities:Internet
        AS Path   :(65002) 65001 4355 2548 3561 5400 6669 5548
        Address: 0x27a4cdb0  Hash:877 (0x03000000) Reference Counts: 2:0:2
...
```

**NOTE:** Portions of this display are truncated for brevity. The purpose of this display is to show all possible fields that might display rather than to show complete output.

***Syntax:*** show ipv6 bgp attribute-entries

For information about display displaying route-attribute entries for a specified BGP4+ neighbor, see “Displaying BGP4+ Neighbor Route-Attribute Entries” on page 8-41.

This display shows the following information:

**Table 8.4: BGP4+ route-attribute entries information**

| This Field... | Displays... |
|---|---|
| Total number of BGP Attribute Entries | The number of entries contained in the router’s BGP4+ route-attribute entries table. |
| Next Hop | The IPv6 address of the next hop router for routes that have this set of attributes. |
| Metric | The cost of the routes that have this set of attributes. |
| Origin | The source of the route information. The origin can be one of the following:<br>• EGP – The routes with this set of attributes came to BGP4+ through EGP.<br>• IGP – The routes with this set of attributes came to BGP4+ through IGP.<br>• INCOMPLETE – The routes came from an origin other than one of the above. For example, they may have been redistributed from OSPFv3 or RIPng.<br>When BGP4+ compares multiple routes to a destination to select the best route, IGP is preferred over EGP, and both are preferred over INCOMPLETE. |
| Originator | The originator of the route in a route-reflector environment. |
| Cluster List | The route-reflector clusters through which this set of attributes has passed. |
| Aggregator | Aggregator information:<br>• AS Number shows the AS in which the network information in the attribute set was aggregated. This value applies only to aggregated routes and is otherwise 0.<br>• Router-ID shows the router that originated this aggregator. |
| Atomic | Whether the network information in this set of attributes has been aggregated *and* this aggregation has resulted in information loss.<br>• TRUE – Indicates information loss has occurred<br>• FALSE – Indicates no information loss has occurred<br>• None – Indicates this attribute is not present.<br>**Note**: Information loss under these circumstances is a normal part of BGP4+ and does not indicate an error. |
| Local Pref | The degree of preference for routes that use this set of attributes relative to other routes in the local AS. |
| Communities | The communities that routes with this set of attributes are in. |

**Table 8.4: BGP4+ route-attribute entries information (Continued)**

| This Field... | Displays... |
|---|---|
| AS Path | The ASs through which routes with this set of attributes have passed. The local AS is shown in parentheses. |
| Address | For debugging purposes only. |
| Hash | For debugging purposes only. |
| Reference Counts | For debugging purposes only. |

## Displaying the BGP4+ Running Configuration

To view the active BGP4+ configuration information contained in the running configuration without displaying the entire running configuration, enter the following command at any level of the CLI:

```
BigIron# show ipv6 bgp config
Current BGP configuration:
router bgp
 local-as 1000
 neighbor peer_group1 peer-group
 neighbor 2001:4383:e0ff:783a::3 remote-as 1001
 neighbor 2001:4484:edd3:8389::1 remote-as 1002
 neighbor 2001:efff:80::23 peer-group peer_group1
 neighbor 2001:efff:80::23 remote-as 1003
 address-family ipv4 unicast
 no neighbor 2001:4383:e0ff:783a::3 activate
 no neighbor 2001:4484:edd3:8389::1 activate
 no neighbor 2001:efff:80::23 activate
 exit-address-family

 address-family ipv4 multicast
 exit-address-family

 address-family ipv6 unicast
 network 3ff0:ec21::/32
 neighbor peer_group1 activate
 neighbor 2001:4484:edd3:8389::1 activate
 exit-address-family

end
```

***Syntax:*** show ipv6 bgp config

## Displaying Dampened BGP4+ Paths

To display BGP4+ paths that have been dampened (suppressed) by route flap dampening, enter the following command at any level of the CLI:

```
BigIron# show ipv6 bgp dampened-paths
Status Code >:best d:damped h:history *:valid
   Network From  Flaps  Since  Reuse  Path
*d 8::/13 2000:1:1::1 1 0 :1 :14 0 :2 :20 100 1002 1000
*d 1::/16 2000:1:1::1 1 0 :1 :14 0 :2 :20 100 1002 1000
*d 4::/14 2000:1:1::1 1 0 :1 :14 0 :2 :20 100 1002 1000
*d 2::/15 2000:1:1::1 1 0 :1 :14 0 :2 :20 100 1002 1000
*d 0:8000::/17 2000:1:1::1 1 0 :1 :14 0 :2 :20 100 1002 1000
*d 2000:1:17::/642000:1:1::1 1 0 :1 :18 0 :2 :20 100
```

***Syntax:*** show ipv6 bgp dampened-paths

This display shows the following information:

**Table 8.5: Dampened BGP4+ path information**

| This Field... | Displays... |
|---|---|
| Status codes | A list of the characters the display uses to indicate the path's status. The status code appears in the left column of the display, to the left of each route. The status codes are described in the command's output. The status column displays a "d" for each dampened route. |
| Network | The destination network of the route. |
| From | The IPv6 address of the advertising peer. |
| Flaps | The number of times the path has flapped. |
| Since | The amount of time (in hh:mm:ss) since the first flap of this route. |
| Reuse | The amount of time (in hh:mm:ss) after which the path is available again. |
| Path | The AS path of the route. |

## Displaying Filtered-Out BGP4+ Routes

When you enable the soft reconfiguration feature, the router saves all updates received from the specified neighbor or peer group. The saved updates include those that contain routes that are filtered out by the BGP4+ route policies in effect on the router. (For more information about soft reconfiguration, see the "Configuring BGP4" chapter in *Foundry Router Configuration Guide*.)

You can display a summary or more detailed information about routes that have been filtered out by BGP4+ route policies.

To display a summary of the routes that have been filtered out by BGP4+ route policies, enter the following command at any level of the CLI:

```
BigIron# show ipv6 bgp filtered-routes
Searching for matching routes, use ^C to quit...
Status A:AGGREGATE B:BEST b:NOT-INSTALLED-BEST C:CONFED_EBGP D:DAMPED
       E:EBGP H:HISTORY I:IBGP L:LOCAL M:MULTIPATH S:SUPPRESSED F:FILTERED
       Prefix             Next Hop        Metric     LocPrf     Weight Status
1      3000::/16          2000:4::110                 100        0        EF
         AS_PATH: 65001 4355 701 80
2      4000::/16          2000:4::110                 100        0        EF
         AS_PATH: 65001 4355 1
3      5000::/16          2000:4::110                 100        0        EF
         AS_PATH: 65001 4355 701 1 189
```

The routes displayed by the command are the routes that the router's BGP policies filtered out. The router did not place the routes in the BGP4+ route table, but did keep the updates. If a policy change causes these routes to be permitted, the router does not need to request the route information from the neighbor, but instead uses the information in the updates.

***Syntax:*** show ipv6 bgp filtered-routes [<ipv6-prefix>/<prefix-length> [longer-prefixes] | [as-path-access-list <name>] | [prefix-list <name>]

The <ipv6-prefix>/<prefix-length> parameter displays the specified IPv6 prefix of the destination network only. You must specify the <ipv6-prefix> parameter in hexadecimal using 16-bit values between colons as documented in RFC 2373. You must specify the <prefix-length> parameter as a decimal value. A slash mark (/) must follow the <ipv6-prefix> parameter and precede the <prefix-length> parameter.

The **longer-prefixes** keyword allows you to display routes that match a specified or longer IPv6 prefix. For example, if you specify **2002::/16 longer-prefixes**, then all routes with the prefix 2002::/16 or that have a longer prefix (such as 2002:e016::/32) are displayed.

The **as-path-access-list** <name> parameter specifies an AS-path ACL. Specify an ACL name. Only the routes permitted by the AS-path ACL are displayed.

The **prefix-list** <name> parameter specifies an IPv6 prefix list. Only the routes permitted by the prefix list are displayed.

This display shows the following information:

**Table 8.6: Summary of filtered-out BGP4+ route information**

| This Field... | Displays... |
|---|---|
| Number of BGP4+ Routes matching display condition | The number of routes that matched the display parameters you entered. This is the number of routes displayed by the command. |
| Status codes | A list of the characters the display uses to indicate the route's status. The status code appears in the left column of the display, to the left of each route. The status codes are described in the command's output. The status column displays an "F" for each filtered route. |
| Prefix | The network address and prefix. |
| Next Hop | The next-hop router for reaching the network from the router. |
| Metric | The value of the route's MED attribute. If the route does not have a metric, this field is blank. |

**Table 8.6: Summary of filtered-out BGP4+ route information (Continued)**

| This Field... | Displays... |
|---|---|
| LocPrf | The degree of preference for this route relative to other routes in the local AS. When the BGP4+ algorithm compares routes on the basis of local preferences, the route with the higher local preference is chosen. The preference can have a value from 0 – 4294967295. |
| Weight | The value that this router associates with routes from a specific neighbor. For example, if the router receives routes to the same destination from two BGP4+ neighbors, the router prefers the route from the neighbor with the larger weight. |
| Status | The route's status, which can be one or more of the following:<br>• A – AGGREGATE – The route is an aggregate route for multiple networks.<br>• B – BEST – BGP4+ has determined that this is the optimal route to the destination.<br>• b – NOT-INSTALLED-BEST – BGP4+ has determined that this is the optimal route to the destination but did not install it in the IPv6 route table because the router received better routes from other sources (such as OSPFv3, RIPng, or static IPv6 routes).<br>• C – CONFED_EBGP – The route was learned from a neighbor in the same confederation and AS, but in a different sub-AS within the confederation.<br>• D – DAMPED – This route has been dampened (by the route dampening feature), and is currently unusable.<br>• E – EBGP – The route was learned through a router in another AS.<br>• H – HISTORY – Route dampening is configured for this route, and the route has a history of flapping and is unreachable now.<br>• I – IBGP – The route was learned through a router in the same AS.<br>• L – LOCAL – The route originated on this router.<br>• M – MULTIPATH – BGP4+ load sharing is enabled and this route was selected as one of the best ones to the destination. The best route among the multiple paths also is marked with "B".<br>**Note**: If the "m" is shown in lowercase, the software was not able to install the route in the IPv6 route table.<br>• S – SUPPRESSED – This route was suppressed during aggregation and thus is not advertised to neighbors.<br>• F – FILTERED – This route was filtered out by BGP4+ route policies on the router, but the router saved updates containing the filtered routes. |

To display detailed information about the routes that have been filtered out by BGP4+ route policies, enter the following command at any level of the CLI:

```
BigIron# show ipv6 bgp filtered-routes detail
Status A:AGGREGATE B:BEST b:NOT-INSTALLED-BEST C:CONFED_EBGP D:DAMPED
E:EBGP H:HISTORY I:IBGP L:LOCAL M:MULTIPATH S:SUPPRESSED F:FILTERED
1  Prefix: 800:2:1::/64, Status: EF, Age: 0h0m10s
   NEXT_HOP: 2000:1:1::1, Learned from Peer: 2000:1:1::1 (100)
   LOCAL_PREF: 100, MED: 0, ORIGIN: incomplete, Weight: 0
   AS_PATH: 100

2  Prefix: 900:1:18::/64, Status: EF, Age: 0h0m10s
   NEXT_HOP: 2000:1:1::1, Learned from Peer: 2000:1:1::1 (100)
   LOCAL_PREF: 100, MED: 0, ORIGIN: incomplete, Weight: 0
   AS_PATH: 100

3  Prefix: 1000:1:1::/64, Status: EF, Age: 0h0m10s
   NEXT_HOP: 2000:1:1::1, Learned from Peer: 2000:1:1::1 (100)
   LOCAL_PREF: 100, MED: 0, ORIGIN: incomplete, Weight: 0
   AS_PATH: 100

4  Prefix: 2000:1:1::/64, Status: EF, Age: 0h0m10s
   NEXT_HOP: 2000:1:1::1, Learned from Peer: 2000:1:1::1 (100)
   LOCAL_PREF: 100, MED: 0, ORIGIN: incomplete, Weight: 0
   AS_PATH: 100

5  Prefix: 2000:1:11::1/128, Status: EF, Age: 0h0m10s
   NEXT_HOP: 2000:1:1::1, Learned from Peer: 2000:1:1::1 (100)
   LOCAL_PREF: 100, MED: 0, ORIGIN: igp, Weight: 0
   AS_PATH: 100

6  Prefix: 2000:1:17::/64, Status: EF, Age: 0h0m10s
   NEXT_HOP: 2000:1:1::1, Learned from Peer: 2000:1:1::1 (100)
   LOCAL_PREF: 100, MED: 0, ORIGIN: incomplete, Weight: 0
   AS_PATH: 100
```

***Syntax:*** show ipv6 bgp filtered-routes detail [<ipv6-prefix>/<prefix-length> [longer-prefixes] | [as-path-access-list <name>] | [prefix-list <name>]

The <ipv6-prefix>/<prefix-length> parameter displays the specified IPv6 prefix of the destination network only. You must specify the <ipv6-prefix> parameter in hexadecimal using 16-bit values between colons as documented in RFC 2373. You must specify the <prefix-length> parameter as a decimal value. A slash mark (/) must follow the <ipv6-prefix> parameter and precede the <prefix-length> parameter.

The **longer-prefixes** keyword allows you to display routes that match a specified or longer IPv6 prefix. For example, if you specify **2002::/16 longer-prefixes**, then all routes with the prefix 2002::/16 or that have a longer prefix (such as 2002:e016::/32) are displayed.

The **as-path-access-list** <name> parameter specifies an AS-path ACL. Only the routes permitted by the AS-path ACL are displayed.

The **prefix-list** <name> parameter specifies an IPv6 prefix list. Only the routes permitted by the prefix list are displayed.

This display shows the following information:

**Table 8.7: Detailed filtered-rut BGP4+ Route information**

| This Field... | Displays... |
|---|---|
| Status codes | A list of the characters the display uses to indicate the route's status. The Status field display an "F" for each filtered route. |
| Prefix | For information about this field, see Table 8.6 on page 8-24. |
| Status | For information about this field, see Table 8.6 on page 8-24. |
| Age | The age of the route, in seconds. |
| Next hop | For information about this field, see Table 8.6 on page 8-24. |
| Learned from peer | The IPv6 address of the neighbor from which this route is learned. "Local Router" indicates that the router itself learned the route. |
| Local pref | For information about this field, see Table 8.6 on page 8-24. |
| MED | The value of the advertised route's MED attribute. If the route does not have a metric, this field is blank. |

**Table 8.7: Detailed filtered-rut BGP4+ Route information (Continued)**

| This Field... | Displays... |
|---|---|
| Origin | The source of the route information. The origin can be one of the following:<br>• A – AGGREGATE – The route is an aggregate route for multiple networks.<br>• B – BEST – BGP4+ has determined that this is the optimal route to the destination.<br>• b – NOT-INSTALLED-BEST – BGP4+ has determined that this is the optimal route to the destination but did not install it in the IPv6 route table because the router received better routes from other sources (such as OSPFv3, RIPng, or static IPv6 routes).<br>• C – CONFED_EBGP – The route was learned from a neighbor in the same confederation and AS, but in a different sub-AS within the confederation.<br>• D – DAMPED – This route has been dampened (by the route dampening feature), and is currently unusable.<br>• E – EBGP – The route was learned through a router in another AS.<br>• H – HISTORY – Route dampening is configured for this route, and the route has a history of flapping and is unreachable now.<br>• I – IBGP – The route was learned through a router in the same AS.<br>• L – LOCAL – The route originated on this router.<br>• M – MULTIPATH – BGP4+ load sharing is enabled and this route was selected as one of the best ones to the destination. The best route among the multiple paths also is marked with “B”.<br>Note: If the “m” is shown in lowercase, the software was not able to install the route in the IPv6 route table.<br>• S – SUPPRESSED – This route was suppressed during aggregation and thus is not advertised to neighbors.<br>• F – FILTERED – This route was filtered out by BGP4+ route policies on the router, but the router saved updates containing the filtered routes. |
| Weight | For information about this field, see Table 8.6 on page 8-24. |
| AS path | The ASs through which routes with this set of attributes have passed. The local AS is shown in parentheses. |

## Displaying Route Flap Dampening Statistics

To display route dampening statistics for all dampened routes, enter the following command at any level of the CLI:

```
BigIron# show ipv6 bgp flap-statistics
Total number of flapping routes: 14
    Status Code  >:best d:damped h:history *:valid
    Network            From            Flaps Since    Reuse    Path
h>  2001:2::/32        3001:23::47      1    0 :0 :13 0 :0 :0  65001 4355 1 701
*>  3892:34::/32       3001:23::47      1    0 :1 :4  0 :0 :0  65001 4355 701 62
```

***Syntax:*** show ipv6 bgp flap-statistics [<ipv6-prefix>/<prefix-length> [longer-prefixes] | as-path-filter <number> | neighbor <ipv6-address> | regular-expression <regular-expression>]

The <ipv6-prefix>/<prefix-length> parameter displays statistics for the specified IPv6 prefix only. You must specify the <ipv6-prefix> parameter in hexadecimal using 16-bit values between colons as documented in RFC 2373. You must specify the <prefix-length> parameter as a decimal value. A slash mark (/) must follow the <ipv6-prefix> parameter and precede the <prefix-length> parameter.

The **longer-prefixes** keyword allows you to display statistics for routes that match a specified or longer IPv6 prefix. For example, if you specify **2000::/16 longer-prefixes**, then all routes with the prefix 2002:: or that have a longer prefix (such as 2002:e016::/32) are displayed.

The **as-path-filter** <number> parameter specifies an AS path filter to display. Specify a filter number.

The **neighbor** <ipv6-address> parameter displays statistics for routes learned from the specified neighbor only. You also can display route flap statistics for routes learned from a neighbor by entering the following command: **show ipv6 bgp neighbor** <ipv6-address> **flap-statistics**.

The **regular-expression** <regular-expression> parameter is a regular expression. The regular expressions are the same ones supported for BGP4 AS-path filters. For more information about regular expressions, see the "Configuring BGP4" chapter in the *Foundry Router Configuration Guide*.

You can also display route flap dampening statistics for a specified IPv6 neighbor. For more information, see "Displaying Route Flap Dampening Statistics for a BGP4+ Neighbor" on page 8-43.

This display shows the following information:

**Table 8.8: Route flap dampening statistics**

| This Field... | Displays... |
|---|---|
| Total number of flapping routes | The total number of routes in the router's BGP4+ route table that have changed state and thus have been marked as flapping routes. |
| Status code | Indicates the dampening status of the route, which can be one of the following:<br>• > – This is the best route among those in the BGP4+ route table to the route's destination.<br>• d – This route is currently dampened, and thus unusable.<br>• h – The route has a history of flapping and is unreachable now.<br>• * – The route has a history of flapping but is currently usable. |
| Network | The destination network of the route. |
| From | The IPv6 address of the advertising peer. |
| Flaps | The number of flaps (state changes) the route has experienced. |

**Table 8.8: Route flap dampening statistics**

| This Field... | Displays... |
|---|---|
| Since | The amount of time (in hh:mm:ss) since the first flap of this route. |
| Reuse | The amount of time (in hh:mm:ss) after which the path is again available. |
| Path | The AS path of the route. |

You also can display all the dampened routes by using the **show ipv6 bgp dampened-paths** command. For more information, see “Displaying Dampened BGP4+ Paths” on page 8-23.

## Displaying BGP4+ Neighbor Information

You can display the following information about a router’s BGP4+ neighbors:

- Configuration information and statistics.
- Router advertisements.
- Route-attribute entries.
- Route flap dampening statistics.
- The last packet containing an error.
- Received Outbound Route Filters (ORFs).
- Routes received from a neighbor.
- BGP4+ Routing Information Base (RIB).
- Received best, not installed best, and unreachable routes.
- Route summary.

## Displaying IPv6 Neighbor Configuration Information and Statistics

To display BGP4+ neighbor configuration information and statistics, enter the following command at any level of the CLI:

```
BigIron# show ipv6 bgp neighbor 2000:4::110
1   IP Address: 2000:4::110, AS: 65002 (EBGP), RouterID: 1.1.1.1
    State: ESTABLISHED, Time: 5d20h38m54s, KeepAliveTime: 60, HoldTime: 180
       RefreshCapability: Received
    Messages:    Open    Update  KeepAlive Notification Refresh-Req
       Sent    : 1       2       8012      0            0
       Received: 1       0       7880      0            0
    Last Update Time: NLRI       Withdraw          NLRI       Withdraw
                  Tx: ---        ---           Rx: ---        ---
    Last Connection Reset Reason:Unknown
    Notification Sent:     Unspecified
    Notification Received: Unspecified
    Neighbor NLRI Negotiation:
      Peer Negotiated IPV6 unicast  capability
      Peer configured for IPV6 unicast  Routes
    TCP Connection state: ESTABLISHED
       Byte Sent:   152411, Received: 149765
       Local host:  2000:4::106, Local  Port: 8222
       Remote host: 2000:4::110, Remote Port: 179
       ISentSeq:  740437769  SendNext:  740590181  TotUnAck:          0
       TotSent:      152412  ReTrans:           0  UnAckSeq:  740590181
       IRcvSeq:   242365900  RcvNext:   242515666  SendWnd:       16384
       TotalRcv:     149766  DupliRcv:          0  RcvWnd:        16384
       SendQue:           0  RcvQue:            0  CngstWnd:       1440
...
```

**NOTE:** Portions of this display are truncated for brevity. The purpose of this display is to show all possible fields that might display rather than to show complete output.

The display shows all the configured parameters for the neighbor. Only the parameters that have values different from their defaults are shown.

In this example, the number in the far left column indicates the neighbor for which information is displayed. When you list information for multiple neighbors, this number makes the display easier to read.

The TCP statistics at the end of the display show status for the TCP session with the neighbor. Most of the fields show information stored in the router's Transmission Control Block (TCB) for the TCP session between the router and its neighbor. These fields are described in detail in section 3.2 of RFC 793, "Transmission Control Protocol Functional Specification".

***Syntax:*** show ipv6 bgp neighbor [<ipv6-address>]

The <ipv6-address> parameter allows you to display information for a specified neighbor only. You must specify the <ipv6-address> parameter in hexadecimal using 16-bit values between colons as documented in RFC 2373.

This display shows the following information:

**Table 8.9: BGP4+ neighbor configuration information and statistics**

| This Field... | Displays... |
|---|---|
| IP Address | The IPv6 address of the neighbor. |

**Table 8.9: BGP4+ neighbor configuration information and statistics (Continued)**

| This Field... | Displays... |
|---|---|
| AS | The AS in which the neighbor resides. |
| EBGP/IBGP | Whether the neighbor session is an IBGP session, an EBGP session, or a confederation EBGP session.<br>• EBGP – The neighbor is in another AS.<br>• EBGP_Confed – The neighbor is a member of another sub-AS in the same confederation.<br>• IBGP – The neighbor is in the same AS. |
| RouterID | The neighbor's router ID. |
| State | The state of the router's session with the neighbor. The states are from the router's perspective of the session, not the neighbor's perspective. The state values can be one of the following:<br>• IDLE – The BGP4+ process is waiting to be started. Usually, enabling BGP4 or establishing a neighbor session starts the BGP4+ process.<br>  • A minus sign (-) indicates that the session has gone down and the software is clearing or removing routes.<br>• ADMND – The neighbor has been administratively shut down.<br>  • A minus sign (-) indicates that the session has gone down and the software is clearing or removing routes.<br>• CONNECT – BGP4+ is waiting for the connection process for the TCP neighbor session to be completed.<br>• ACTIVE – BGP4+ is waiting for a TCP connection from the neighbor.<br>**Note**: If the state frequently changes between CONNECT and ACTIVE, there may be a problem with the TCP connection.<br>• OPEN SENT – BGP4+ is waiting for an Open message from the neighbor.<br>• OPEN CONFIRM – BGP4+4 has received an OPEN message from the neighbor and is now waiting for either a KEEPALIVE or NOTIFICATION message. If the router receives a KEEPALIVE message from the neighbor, the state changes to Established. If the message is a NOTIFICATION, the state changes to Idle.<br>• ESTABLISHED – BGP4+ is ready to exchange UPDATE messages with the neighbor.<br>  • If there is more BGP data in the TCP receiver queue, a plus sign (+) is also displayed.<br>**Note**: If you display information for the neighbor using the **show ipv6 bgp neighbor** <ipv6-address> command, the TCP receiver queue value will be greater than 0. |
| Time | The amount of time this session has been in its current state. |

**Table 8.9: BGP4+ neighbor configuration information and statistics (Continued)**

| This Field... | Displays... |
|---|---|
| KeepAliveTime | The keep alive time, which specifies how often this router sends keep alive messages to the neighbor. For information about configuring this parameter, see the "Configuring BGP4" chapter in *Foundry Enterprise Configuration and Management Guide*. |
| HoldTime | The hold time, which specifies how many seconds the router will wait for a KEEPALIVE or UPDATE message from a BGP4+ neighbor before deciding that the neighbor is dead. For information about configuring this parameter, see the "Configuring BGP4" chapter in *Foundry Enterprise Configuration and Management Guide*. |
| RefreshCapability | Whether the router has received confirmation from the neighbor that the neighbor supports the dynamic refresh capability. |
| Messages Sent and Received | The number of messages this router has sent to and received from the neighbor. The display shows statistics for the following message types:<br>• Open<br>• Update<br>• KeepAlive<br>• Notification<br>• Refresh-Req |
| Last Update Time | Lists the last time updates were sent and received for the following:<br>• NLRIs<br>• Withdraws |

**Table 8.9: BGP4+ neighbor configuration information and statistics (Continued)**

| This Field... | Displays... |
|---|---|
| Last Connection Reset Reason | The reason the previous session with this neighbor ended. The reason can be one of the following:<br>• No abnormal error has occurred.<br>• Reasons described in the BGP specifications:<br>  • Message Header Error<br>  • Connection Not Synchronized<br>  • Bad Message Length<br>  • Bad Message Type<br>  • OPEN Message Error<br>  • Unsupported Version Number<br>  • Bad Peer AS Number<br>  • Bad BGP Identifier<br>  • Unsupported Optional Parameter<br>  • Authentication Failure<br>  • Unacceptable Hold Time<br>  • Unsupported Capability<br>  • UPDATE Message Error<br>  • Malformed Attribute List<br>  • Unrecognized Well-known Attribute<br>  • Missing Well-known Attribute<br>  • Attribute Flags Error<br>  • Attribute Length Error<br>  • Invalid ORIGIN Attribute<br>  • Invalid NEXT_HOP Attribute<br>  • Optional Attribute Error<br>  • Invalid Network Field<br>  • Malformed AS_PATH<br>  • Hold Timer Expired<br>  • Finite State Machine Error<br>  • Rcv Notification |

**Table 8.9: BGP4+ neighbor configuration information and statistics (Continued)**

| This Field... | Displays... |
|---|---|
| Last Connection Reset Reason (cont.) | • Reasons specific to the Foundry implementation:<br>  • Reset All Peer Sessions<br>  • User Reset Peer Session<br>  • Port State Down<br>  • Peer Removed<br>  • Peer Shutdown<br>  • Peer AS Number Change<br>  • Peer AS Confederation Change<br>  • TCP Connection KeepAlive Timeout<br>  • TCP Connection Closed by Remote<br>  • TCP Data Stream Error Detected |

**Table 8.9: BGP4+ neighbor configuration information and statistics (Continued)**

| This Field... | Displays... |
|---|---|
| Notification Sent | If the router receives a NOTIFICATION message from the neighbor, the message contains an error code corresponding to one of the following errors. Some errors have subcodes that clarify the reason for the error. Where applicable, the subcode messages are listed underneath the error code messages.<br>• Message Header Error<br>  • Connection Not Synchronized<br>  • Bad Message Length<br>  • Bad Message Type<br>  • Unspecified<br>• Open Message Error<br>  • Unsupported Version<br>  • Bad Peer As<br>  • Bad BGP Identifier<br>  • Unsupported Optional Parameter<br>  • Authentication Failure<br>  • Unacceptable Hold Time<br>  • Unspecified<br>• Update Message Error<br>  • Malformed Attribute List<br>  • Unrecognized Attribute<br>  • Missing Attribute<br>  • Attribute Flag Error<br>  • Attribute Length Error<br>  • Invalid Origin Attribute<br>  • Invalid NextHop Attribute<br>  • Optional Attribute Error<br>  • Invalid Network Field<br>  • Malformed AS Path<br>  • Unspecified<br>• Hold Timer Expired<br>• Finite State Machine Error<br>• Cease<br>• Unspecified |
| Notification Received | See above. |

**Table 8.9: BGP4+ neighbor configuration information and statistics (Continued)**

| This Field... | Displays... |
|---|---|
| Neighbor NLRI Negotiation | The state of the router's NLRI negotiation with the neighbor. The states can include the following:<br>• Peer negotiated IPv6 unicast capability.<br>• Peer configured for IPv6 unicast routes.<br>• Peer negotiated IPv4 unicast capability.<br>• Peer negotiated IPv4 multicast capability. |
| TCP Connection state | The state of the connection with the neighbor. The connection can have one of the following states:<br>• LISTEN – Waiting for a connection request.<br>• SYN-SENT – Waiting for a matching connection request after having sent a connection request.<br>• SYN-RECEIVED – Waiting for a confirming connection request acknowledgment after having both received and sent a connection request.<br>• ESTABLISHED – Data can be sent and received over the connection. This is the normal operational state of the connection.<br>• FIN-WAIT-1 – Waiting for a connection termination request from the remote TCP, or an acknowledgment of the connection termination request previously sent.<br>• FIN-WAIT-2 – Waiting for a connection termination request from the remote TCP.<br>• CLOSE-WAIT – Waiting for a connection termination request from the local user.<br>• CLOSING – Waiting for a connection termination request acknowledgment from the remote TCP.<br>• LAST-ACK – Waiting for an acknowledgment of the connection termination request previously sent to the remote TCP (which includes an acknowledgment of its connection termination request).<br>• TIME-WAIT – Waiting for enough time to pass to be sure the remote TCP received the acknowledgment of its connection termination request.<br>• CLOSED – There is no connection state. |
| Byte Sent | The number of bytes sent. |
| Byte Received | The number of bytes received. |
| Local host | The IPv6 address of the router. |
| Local port | The TCP port the router is using for the BGP4+ TCP session with the neighbor. |
| Remote host | The IPv6 address of the neighbor. |

**Table 8.9: BGP4+ neighbor configuration information and statistics (Continued)**

| This Field... | Displays... |
|---|---|
| Remote port | The TCP port the neighbor is using for the BGP4+ TCP session with the router. |
| ISentSeq | The initial send sequence number for the session. |
| SendNext | The next sequence number to be sent. |
| TotUnAck | The number of sequence numbers sent by the router that have not been acknowledged by the neighbor. |
| TotSent | The number of sequence numbers sent to the neighbor. |
| ReTrans | The number of sequence numbers that the router retransmitted because they were not acknowledged. |
| UnAckSeq | The current acknowledged sequence number. |
| IRcvSeq | The initial receive sequence number for the session. |
| RcvNext | The next sequence number expected from the neighbor. |
| SendWnd | The size of the send window. |
| TotalRcv | The number of sequence numbers received from the neighbor. |
| DupliRcv | The number of duplicate sequence numbers received from the neighbor. |
| RcvWnd | The size of the receive window. |
| SendQue | The number of sequence numbers in the send queue. |
| RcvQue | The number of sequence numbers in the receive queue. |
| CngstWnd | The number of times the window has changed. |

## Displaying Routes Advertised to a BGP4+ Neighbor

You can display a summary or detailed information about the following:

- All routes a router has advertised to a neighbor.
- A specified route a router has advertised to a neighbor.

For example, to display a summary of all routes a router has advertised to neighbor 2000:4::110, enter the following command at any level of the CLI:

```
BigIron# show ipv6 bgp neighbor 2000:4::110 advertised-routes
       There are 2 routes advertised to neighbor 2000:4::110
Status A:AGGREGATE B:BEST b:NOT-INSTALLED-BEST E:EBGP I:IBGP L:LOCAL
       Prefix             Next Hop        Metric     LocPrf     Weight Status
1      2002:1234::/32     ::              1                     32768  BL
         AS_PATH:
2      2002::/16          ::              1                     32768  BL
         AS_PATH:
```

***Syntax:*** show ipv6 bgp neighbor <ipv6-address> advertised-routes [detail] <ipv6-prefix>/<prefix-length>

The <ipv6-address> parameter displays routes advertised to a specified neighbor. You must specify this address in hexadecimal using 16-bit values between colons as documented in RFC 2373.

The **detail** keyword displays detailed information about the advertised routes. If you do not specify this keyword, a summary of the advertised routes displays.

The <ipv6-prefix>/<prefix-length> parameter displays the specified route advertised to the neighbor only. You must specify the <ipv6-prefix> parameter in hexadecimal using 16-bit values between colons as documented in RFC 2373. You must specify the <prefix-length> parameter as a decimal value. A slash mark (/) must follow the <ipv6-prefix> parameter and precede the <prefix-length> parameter.

This display shows the following information:

**Table 8.10: Summary of route information advertised to a BGP4+ neighbor**

| This Field... | Displays... |
|---|---|
| Number of BGP4+ Routes advertised to specified neighbor (appears only in display for all routes) | The number of routes displayed by the command. |
| Status codes | A list of the characters the display uses to indicate the route's status. The status code appears in the Status column of the display. The status codes are described in the command's output. |
| Prefix | The advertised route's prefix. |
| Next Hop | The next-hop router for reaching the advertised route from the router. |
| Metric | The value of the advertised route's MED attribute. If the route does not have a metric, this field is blank. |
| LocPrf | The degree of preference for the advertised route relative to other routes in the local AS. When the BGP4+ algorithm compares routes on the basis of local preferences, the route with the higher local preference is chosen. The preference can have a value from 0 – 4294967295. |
| Weight | The value that this router associates with routes from a specific neighbor. For example, if the router receives routes to the same destination from two BGP4+ neighbors, the router prefers the route from the neighbor with the larger weight. |
| Status | The advertised route's status, which can be one or more of the following:<br>• A – AGGREGATE. The route is an aggregate route for multiple networks.<br>• B – BEST. BGP4+ has determined that this is the optimal route to the destination.<br>• b – NOT-INSTALLED-BEST – BGP4+ has determined that this is the optimal route to the destination but did not install it in the IPv6 route table because the router received better routes from other sources (such as OSPFv3, RIPng, or static IPv6 routes).<br>• E – EBGP. The route was learned through a router in another AS.<br>• I – IBGP. The route was learned through a router in the same AS.<br>• L – LOCAL. The route originated on this router. |
| AS-PATH | The AS-path information for the route. |

For example, to display details about all routes a router has advertised to neighbor 2000:4::110, enter the following command at any level of the CLI:

```
BigIron# show ipv6 bgp neighbor 2000:4::110 advertised-routes detail
There are 2 routes advertised to neighbor 2000:4::110
Status A:AGGREGATE B:BEST b:NOT-INSTALLED-BEST E:EBGP I:IBGP L:LOCAL
1       Prefix: 2002:1234::/32,  Status: BL,  Age: 6d13h28m7s
         NEXT_HOP: 2000:4::106,  Learned from Peer: Local Router
          LOCAL_PREF: none,  MED: 1,  ORIGIN: incomplete,  Weight: 32768
           AS_PATH:
            Adj_RIB_out count: 1,  Admin distance 190
2       Prefix: 2002::/16,  Status: BL,  Age: 6d13h31m22s
         NEXT_HOP: 2000:4::106,  Learned from Peer: Local Router
          LOCAL_PREF: none,  MED: 1,  ORIGIN: incomplete,  Weight: 32768
           AS_PATH:
```

This display shows the following information:

**Table 8.11: Detailed route information advertised to a BGP4+ neighbor**

| This Field... | Displays... |
|---|---|
| Number of BGP4+ Routes advertised to specified neighbor (appears only in display for all routes) | For information about this field, see Table 8.10 on page 8-39. |
| Status codes | For information about this field, see Table 8.10 on page 8-39. |
| Prefix | For information about this field, see Table 8.10 on page 8-39. |
| Status | For information about this field, see Table 8.10 on page 8-39. |
| Age | The age of the advertised route, in seconds. |
| Next Hop | For information about this field, see Table 8.10 on page 8-39. |
| Learned from Peer | The IPv6 address of the neighbor from which this route is learned. “Local Router” indicates that the router itself learned the route. |
| LOCAL_PREF | For information about this field, see Table 8.10 on page 8-39. |
| MED | The value of the advertised route’s MED attribute. If the route does not have a metric, this field is blank. |
| Origin | The source of the route information. The origin can be one of the following:<br>• EGP – The routes with this set of attributes came to BGP4+ through EGP.<br>• IGP – The routes with this set of attributes came to BGP4+ through IGP.<br>• INCOMPLETE – The routes came from an origin other than one of the above. For example, they may have been redistributed from OSPFv3 or RIPng.<br>When BGP4+ compares multiple routes to a destination to select the best route, IGP is preferred over EGP and both are preferred over INCOMPLETE. |

**Table 8.11: Detailed route information advertised to a BGP4+ neighbor (Continued)**

| This Field... | Displays... |
|---|---|
| Weight | For information about this field, see Table 8.10 on page 8-39. |
| AS-PATH | The AS-path information for the route. |
| Adj RIB out count | The number of routes in the router's current BGP4+ Routing Information Base (Adj-RIB-Out) for a specified neighbor. |
| Admin distance | The administrative distance of the route. |

## Displaying BGP4+ Neighbor Route-Attribute Entries

The route-attribute entries table lists sets of BGP4+ attributes stored in the router's memory. Each set of attributes is unique and can be associated with one or more routes. In fact, the router typically has fewer route attribute entries than routes.

For example, to display the route-attribute entries table for a BGP4+ neighbor 2000:4::110, enter the following command:

```
BigIron# show ipv6 bgp neighbor 2000:4::110 attribute-entries
                Total number of BGP Attribute Entries: 1
1       Next Hop  :2000:4::106        Metric   :1              Origin:INCOMP
        Originator:0.0.0.0            Cluster List:None
        Aggregator:AS Number :0      Router-ID:0.0.0.0         Atomic:None
        Local Pref:100               Communities:Internet
        AS Path   :65001
        Address: 0x26579354  Hash:332 (0x0301fcd4) Reference Counts: 2:0:0
```

***Syntax:*** show ipv6 bgp neighbor <ipv6-address> attribute-entries

The <ipv6-address> parameter displays the route attribute entries for a specified neighbor. You must specify this address in hexadecimal using 16-bit values between colons as documented in RFC 2373.

This display shows the following information:

**Table 8.12: BGP4+ neighbor route-attribute entries information**

| This Field... | Displays... |
|---|---|
| Total number of BGP Attribute Entries | The number of route attribute entries for the specified neighbor. |
| Next Hop | The IPv6 address of the next hop router for routes that have this set of attributes. |
| Metric | The cost of the routes that have this set of attributes. |

**Table 8.12: BGP4+ neighbor route-attribute entries information (Continued)**

| This Field... | Displays... |
|---|---|
| Origin | The source of the route information. The origin can be one of the following:<br>• EGP – The routes with this set of attributes came to BGP4+ through EGP.<br>• IGP – The routes with this set of attributes came to BGP4+ through IGP.<br>• INCOMPLETE – The routes came from an origin other than one of the above. For example, they may have been redistributed from OSPFv3 or RIPng.<br>When BGP4+ compares multiple routes to a destination to select the best route, IGP is preferred over EGP and both are preferred over INCOMPLETE. |
| Originator | The originator of the route in a route reflector environment. |
| Cluster List | The route-reflector clusters through which this set of attributes has passed. |
| Aggregator | Aggregator information:<br>• `AS Number` shows the AS in which the network information in the attribute set was aggregated. This value applies only to aggregated routes and is otherwise 0.<br>• `Router-ID` shows the router that originated this aggregator. |
| Atomic | Whether the network information in this set of attributes has been aggregated *and* this aggregation has resulted in information loss.<br>• TRUE – Indicates information loss has occurred<br>• FALSE – Indicates no information loss has occurred<br>• None – Indicates the attribute is not present.<br>**Note**: Information loss under these circumstances is a normal part of BGP4+ and does not indicate an error. |
| Local Pref | The degree of preference for routes that use this set of attributes relative to other routes in the local AS. |
| Communities | The communities that routes with this set of attributes are in. |
| AS Path | The ASs through which routes with this set of attributes have passed. The local AS is shown in parentheses. |
| Address | For debugging purposes only. |
| Hash | For debugging purposes only. |
| Reference Counts | For debugging purposes only. |

### Displaying Route Flap Dampening Statistics for a BGP4+ Neighbor

To display route flap dampening statistics for a specified BGP4+ neighbor, enter the following command at any level of the CLI:

```
BigIron# show ipv6 bgp neighbor 2000:4::110 flap-statistics
Total number of flapping routes: 14
    Status Code  >:best d:damped h:history *:valid
    Network            From            Flaps Since    Reuse    Path
h>  2001:2::/32        166.90.213.77   1     0 :0 :13 0 :0 :0  65001 4355 1 701
*>  3892:34::/32       166.90.213.77   1     0 :1 :4  0 :0 :0  65001 4355 701 62
```

***Syntax:*** show ipv6 bgp neighbor <ipv6-address> flap-statistics

The <ipv6-address> parameter displays the route flap dampening statistics for a specified neighbor. You must specify this address in hexadecimal using 16-bit values between colons as documented in RFC 2373.

This display shows the following information:

**Table 8.13: Route flap dampening statistics for a BGP4+ neighbor**

| This Field... | Displays... |
|---|---|
| Total number of flapping routes | The total number of routes in the neighbor's BGP4+ route table that have changed state and thus have been marked as flapping routes. |
| Status code | Indicates the status of the route, which can be one of the following:<br>• > – This is the best route among those in the neighbor's BGP4+ route table to the route's destination.<br>• d – This route is currently dampened, and thus unusable.<br>• h – The route has a history of flapping and is unreachable now.<br>• * – The route has a history of flapping but is currently usable. |
| Network | The destination network of the route. |
| From | The IPv6 address of the advertising peer. |
| Flaps | The number of flaps (state changes) the route has experienced. |
| Since | The amount of time (in hh:mm:ss) since the first flap of this route. |
| Reuse | The amount of time (in hh:mm:ss) after which the path is again available. |
| Path | The AS path of the route. |

You also can display all the dampened routes by using the **show ipv6 bgp dampened-paths** command. For more information, see "Displaying Dampened BGP4+ Paths" on page 8-23.

### Displaying Last Error Packet from a BGP4+ Neighbor

You can display information about the last packet that contained an error from any of a router's neighbors. The displayed information includes the error packet's contents decoded in a human-readable format.

For example, to display information about the last error packet from any of a router's neighbors, enter the following command:

```
BigIron# show ipv6 bgp neighbor last-packet-with-error
    Total number of BGP Neighbors: 266
No received packet with error logged for any neighbor
```

***Syntax:*** show ipv6 bgp neighbor last-packet-with-error

This display shows the following information:

**Table 8.14: Last error packet information for BGP4+ neighbors**

| This Field... | Displays... |
|---|---|
| Total number of BGP Neighbors | The total number of configured neighbors for a router. |
| Last error | The error packet's contents decoded in a human-readable format or notification that no packets with an error were received. |

### Displaying Outbound Route Filters Received from a BGP4+ Neighbor

You can display the Outbound Route Filters (ORFs) received from a BGP4+ neighbor. This option applies to cooperative route filtering feature. For more information about this feature, see the "Configuring BGP4" chapter in the *Foundry Router Configuration Guide*.

For example, to display the ORFs received from neighbor 2000:2::110, enter the following command:

```
BigIron# show ipv6 bgp neighbor 2000:2::110 received prefix-filter
ip prefix-list 2000:2::110: 4 entries
     seq 5 permit 3000:3::45/16 ge 18 le 28
     seq 10 permit 4000:4::88/24
     seq 15 permit 5000:5::37/8 le 32
     seq 20 permit 6000:6::83/16 ge 18
```

***Syntax:*** show ipv6 bgp neighbor <ipv6-address> received prefix-filter

The <ipv6-address> parameter displays the prefix filter learned from a specified neighbor. You must specify this address in hexadecimal using 16-bit values between colons as documented in RFC 2373.

### Displaying Routes Received from a BGP4+ Neighbor

You can display a summary or detailed route information received in route updates from a specified BGP4+ neighbor since you enabled the soft reconfiguration feature. For more information about soft reconfiguration, see the "Configuring BGP4" chapter in the *Foundry Router Configuration Guide*.

For example, to display a summary of the route information received in route updates from neighbor 2000:4::10, enter the following command at any level of the CLI:

```
BigIron# show ipv6 bgp neighbor 2:2:2:2:: received-routes
There are 4 received routes from neighbor 2:2:2:2::
Searching for matching routes, use ^C to quit...
Status A:AGGREGATE B:BEST b:NOT-INSTALLED-BEST C:CONFED_EBGP D:DAMPED
E:EBGP H:HISTORY I:IBGP L:LOCAL M:MULTIPATH S:SUPPRESSED F:FILTERED
    Prefix     Next Hop    Metric   LocPrf  Weight Status
1   2002::/64  2:2:2:2::   0        100     0      BE
AS_PATH: 400
2   2003::/64  2:2:2:2::   1        100     0      BE
AS_PATH: 400
3   2004::/64  2:2:2:2::   1        100     0      BE
AS_PATH: 400
4   2005::/64  2:2:2:2::   1        100     0      BE
AS_PATH: 400
```

***Syntax:*** show ipv6 bgp neighbor <ipv6-address> received-routes [detail]

The <ipv6-address> parameter displays route information received from a specified neighbor. You must specify this address in hexadecimal using 16-bit values between colons as documented in RFC 2373.

The **detail** keyword displays detailed route information. If you do not specify this parameter, a summary of route information displays.

This display shows the following information:

**Table 8.15: Summary of route information received from a BGP4+ neighbor**

| This Field... | Displays... |
|---|---|
| Number of BGP4+ Routes received from a neighbor | The number of routes displayed by the command. |
| Status codes | A list of the characters the display uses to indicate the route's status. The status code appears in the Status column of the display. The status codes are described in the command's output. |
| Prefix | The received route's prefix. |
| Next Hop | The IPv6 address of the next router that is used when forwarding a packet to the received route. |
| Metric | The value of the route's MED attribute. If the route does not have a metric, this field is blank. |
| LocPrf | The degree of preference for the advertised route relative to other routes in the local AS. When the BGP4+ algorithm compares routes on the basis of local preferences, the route with the higher local preference is chosen. The preference can have a value from 0 – 4294967295. |
| Weight | The value that this router associates with routes from a specific neighbor. For example, if the router receives routes to the same destination from two BGP4+ neighbors, the router prefers the route from the neighbor with the larger weight. |

**Table 8.15: Summary of route information received from a BGP4+ neighbor (Continued)**

| This Field... | Displays... |
|---|---|
| Status | The advertised route's status, which can be one or more of the following:<br>A – AGGREGATE. The route is an aggregate route for multiple networks.<br>B – BEST. BGP4+ has determined that this is the optimal route to the destination.<br>b – NOT-INSTALLED-BEST – BGP4+ has determined that this is the optimal route to the destination but did not install it in the IPv6 route table because the router received better routes from other sources (such as OSPFv3, RIPng, or static IPv6 routes).<br>D – DAMPED. This route has been dampened (by the route dampening feature), and is currently unusable.<br>E – EBGP. The route was learned through a router in another AS.<br>H – HISTORY. Route dampening is configured for this route, and the route has a history of flapping and is unreachable now.<br>I – IBGP. The route was learned through a router in the same AS.<br>L – LOCAL. The route originated on this router.<br>M – MULTIPATH. BGP4+ load sharing is enabled and this route was selected as one of the best ones to the destination. The best route among the multiple paths also is marked with "B".<br>Note: If the "m" is shown in lowercase, the software was not able to install the route in the IPv6 route table.<br>S – SUPPRESSED. This route was suppressed during aggregation and thus is not advertised to neighbors.<br>F – FILTERED. This route was filtered out by BGP4+ route policies on the router, but the router saved updates containing the filtered routes. |

For example, to display details about routes received from neighbor 2000:1:1::1, enter the following command at any level of the CLI:

```
BigIron# show ipv6 bgp neighbor 2000:1:1::1 received-routes detail
There are 4 received routes from neighbor 2000:1:1::1
Searching for matching routes, use ^C to quit...

Status A:AGGREGATE B:BEST b:NOT-INSTALLED-BEST C:CONFED_EBGP D:DAMPED
E:EBGP H:HISTORY I:IBGP L:LOCAL M:MULTIPATH S:SUPPRESSED F:FILTERED

1 Prefix: 1000:1:1::/64, Status: BI, Age: 0h17m25s
NEXT_HOP: 2000:1:1::1, Learned from Peer: 2000:1:1::1 (100)
LOCAL_PREF: 100, MED: 0, ORIGIN: incomplete, Weight: 0
AS_PATH:
Adj_RIB_out count: 1, Admin distance 200

2 Prefix: 2000:1:1::/64, Status: I, Age: 0h17m25s
NEXT_HOP: 2000:1:1::1, Learned from Peer: 2000:1:1::1 (100)
LOCAL_PREF: 100, MED: 0, ORIGIN: incomplete, Weight: 0
AS_PATH:

3 Prefix: 2000:1:11::1/128, Status: BI, Age: 0h17m25s
NEXT_HOP: 2000:1:1::1, Learned from Peer: 2000:1:1::1 (100)
LOCAL_PREF: 100, MED: 0, ORIGIN: igp, Weight: 0
AS_PATH:
Adj_RIB_out count: 1, Admin distance 200

4 Prefix: 2000:1:17::/64, Status: BI, Age: 0h17m25s
NEXT_HOP: 2000:1:1::1, Learned from Peer: 2000:1:1::1 (100)
LOCAL_PREF: 100, MED: 0, ORIGIN: incomplete, Weight: 0
AS_PATH:
Adj_RIB_out count: 1, Admin distance 200
```

This display shows the following information:

**Table 8.16: Detailed route information received from a BGP4+ neighbor**

| This Field... | Displays... |
|---|---|
| Number of BGP4+ routes received from a neighbor | For information about this field, see Table 8.15 on page 8-45. |
| Status codes | For information about this field, see Table 8.15 on page 8-45. |
| Prefix | For information about this field, see Table 8.15 on page 8-45. |
| Status | For information about this field, see Table 8.15 on page 8-45. |
| Age | The age of the route, in seconds. |
| Next hop | The next-hop router for reaching the route from the router. |
| Learned from peer | The IPv6 address of the neighbor from which this route is learned. “Local Router” indicates that the router itself learned the route. |
| Local pref | For information about this field, see Table 8.15 on page 8-45. |
| MED | The value of the route’s MED attribute. If the route does not have a metric, this field is blank. |

**Table 8.16: Detailed route information received from a BGP4+ neighbor (Continued)**

| This Field... | Displays... |
|---|---|
| Origin | The source of the route information. The origin can be one of the following:<br>• EGP – The routes with this set of attributes came to BGP4+ through EGP.<br>• IGP – The routes with this set of attributes came to BGP4+ through IGP.<br>• INCOMPLETE – The routes came from an origin other than one of the above. For example, they may have been redistributed from OSPFv3 or RIPng.<br>When BGP4+ compares multiple routes to a destination to select the best route, IGP is preferred over EGP and both are preferred over INCOMPLETE. |
| Weight | For information about this field, see Table 8.15 on page 8-45. |
| AS Path | For information about this field, see Table 8.15 on page 8-45. |
| Adj RIB out count | The number of routes in the router's current BGP4+ Routing Information Base (Adj-RIB-Out) for a specified neighbor. |
| Admin distance | The administrative distance of the route. |

## Displaying the Adj-RIB-Out for a BGP4+ Neighbor

You can display a summary or detailed information about the following:

- All routes in a router's current BGP4+ Routing Information Base (Adj-RIB-Out) for a specified neighbor.
- A specified route in a router's current BGP4+ RIB for a specified neighbor.

The RIB contains the routes that the router either has most recently sent to the neighbor or is about to send to the neighbor.

For example, to display a summary of all routes in a router's RIB for neighbor 2000:4::110, enter the following command at any level of the CLI:

```
BigIron# show ipv6 bgp neighbor 2000:4::110 rib-out-routes
       There are 2 RIB_out routes for neighbor 2000:4::110
Status A:AGGREGATE B:BEST b:NOT-INSTALLED-BEST E:EBGP I:IBGP L:LOCAL
       Prefix             Next Hop        Metric     LocPrf     Weight Status
1      2002:1234::/32     ::              1          100        32768  BL
         AS_PATH:
2      2002::/16          ::              1          100        32768  BL
         AS_PATH:
```

***Syntax:*** show ipv6 bgp neighbor <ipv6-address> rib-out-routes [<ipv6-prefix>/<prefix-length> | detail [<ipv6-prefix>/<prefix-length> <network-mask>]]

The <ipv6-address> parameter displays the RIB routes for a specified neighbor. You must specify this address in hexadecimal using 16-bit values between colons as documented in RFC 2373.

The <ipv6-prefix>/<prefix-length> parameter displays the specified RIB route for the neighbor. You must specify the <ipv6-prefix> parameter in hexadecimal using 16-bit values between colons as documented in RFC 2373. You must specify the <prefix-length> parameter as a decimal value. A slash mark (/) must follow the <ipv6-prefix> parameter and precede the <prefix-length> parameter.

The **detail** <ipv6-prefix>/<prefix-length> <network-mask> parameter displays detailed information about the specified RIB routes. If you do not specify this parameter, a summary of the RIB routes displays. You must specify the <ipv6-prefix> parameter in hexadecimal using 16-bit values between colons as documented in RFC 2373. You must specify the <prefix-length> parameter as a decimal value. A slash mark (/) must follow the <ipv6-prefix> parameter and precede the <prefix-length> parameter. You must specify the <network-mask> parameter using 8-bit values in dotted decimal notation.

This display shows the following information:

**Table 8.17: Summary of RIB route information for a BGP4+ neighbor**

| This Field... | Displays... |
|---|---|
| Number of RIB_out routes for a specified neighbor (appears only in display for all RIB routes) | The number of RIB routes displayed by the command. |
| Status codes | A list of the characters the display uses to indicate the route's status. The status code appears in the Status column of the display. The status codes are described in the command's output. |
| Prefix | The RIB route's prefix. |
| Next Hop | The next-hop router for reaching the route from the router. |
| Metric | The value of the advertised route's MED attribute. If the route does not have a metric, this field is blank. |
| LocPrf | The degree of preference for the route relative to other routes in the local AS. When the BGP4+ algorithm compares routes on the basis of local preferences, the route with the higher local preference is chosen. The preference can have a value from 0 – 4294967295. |
| Weight | The value that this router associates with routes from a specific neighbor. For example, if the router receives routes to the same destination from two BGP4+ neighbors, the router prefers the route from the neighbor with the larger weight. |
| Status | The RIB route's status, which can be one or more of the following:<br>• A – AGGREGATE. The route is an aggregate route for multiple networks.<br>• B – BEST. BGP4+ has determined that this is the optimal route to the destination.<br>E – EBGP. The route was learned through a router in another AS.<br>• I – IBGP. The route was learned through a router in the same AS.<br>• L – LOCAL. The route originated on this router. |
| AS-PATH | The AS-path information for the route. |

For example, to display details about all RIB routes for neighbor 2000:4::110, enter the following command at any level of the CLI:

```
BigIron# show ipv6 bgp neighbor 2000:4::110 rib-out-routes detail
                    There are 2 RIB_out routes for neighbor 2000:4::110
Status A:AGGREGATE B:BEST b:NOT-INSTALLED-BEST E:EBGP I:IBGP L:LOCAL
1       Prefix: 2002:1234::/32,  Status: BL,  Age: 6d18h17m53s
         NEXT_HOP: ::,  Learned from Peer: Local Router
          LOCAL_PREF: 100,  MED: 1,  ORIGIN: incomplete,  Weight: 32768
           AS_PATH:
            Adj_RIB_out count: 1,  Admin distance 190
2       Prefix: 2002::/16,  Status: BL,  Age: 6d18h21m8s
         NEXT_HOP: ::,  Learned from Peer: Local Router
          LOCAL_PREF: 100,  MED: 1,  ORIGIN: incomplete,  Weight: 32768
           AS_PATH:
```

This display shows the following information:

**Table 8.18: Detailed RIB route information for a BGP4+ neighbor**

| This Field... | Displays... |
|---|---|
| Number of RIB_out routes for a specified neighbor (appears only in display for all routes) | For information about this field, see Table 8.17 on page 8-49. |
| Status codes | For information about this field, see Table 8.17 on page 8-49. |
| Prefix | For information about this field, see Table 8.17 on page 8-49. |
| Status | For information about this field, see Table 8.17 on page 8-49. |
| Age | The age of the RIB route, in seconds. |
| Next Hop | For information about this field, see Table 8.17 on page 8-49. |
| Learned from Peer | The IPv6 address of the neighbor from which this route is learned. “Local Router” indicates that the router itself learned the route. |
| LOCAL_PREF | For information about this field, see Table 8.17 on page 8-49. |
| MED | The value of the RIB route’s MED attribute. If the route does not have a metric, this field is blank. |
| Origin | The source of the route information. The origin can be one of the following:<br>• EGP – The routes with this set of attributes came to BGP4+ through EGP.<br>• IGP – The routes with this set of attributes came to BGP4+ through IGP.<br>• INCOMPLETE – The routes came from an origin other than one of the above. For example, they may have been redistributed from OSPFv3 or RIPng.<br>When BGP4+ compares multiple routes to a destination to select the best route, IGP is preferred over EGP and both are preferred over INCOMPLETE. |
| Weight | For information about this field, see Table 8.17 on page 8-49. |
| AS-PATH | For information about this field, see Table 8.17 on page 8-49. |

### Displaying the Best and Unreachable Routes Received from a BGP4+ Neighbor

You can display a summary or detailed information about the following types of BGP4+ routes received from a specified neighbor:

- Best routes – The “best” routes to their destinations, which are installed in the router’s IPv6 route table.
- Unreachable – The routes whose destinations are unreachable using any of the BGP4+ paths in the IPv6 route table.

For example, to display a summary of the best routes to a destination received from neighbor 2000:4::106, enter the following command:

```
BigIron# show ipv6 bgp neighbor 2000:4::106 routes best
       There are 2 accepted routes from neighbor 2000:4::106
Searching for matching routes, use ^C to quit...
Status A:AGGREGATE B:BEST b:NOT-INSTALLED-BEST C:CONFED_EBGP D:DAMPED
       E:EBGP H:HISTORY I:IBGP L:LOCAL M:MULTIPATH S:SUPPRESSED F:FILTERED
       Prefix             Next Hop        Metric     LocPrf     Weight Status
1      2002::/16          2000:4::106     1          100        0      BE
         AS_PATH: 65001
2      2002:1234::/32     2000:4::106     1          100        0      BE
         AS_PATH: 65001
```

***Syntax:*** show ipv6 bgp neighbor <ipv6-address> routes best | detail [best | unreachable] | unreachable

The <ipv6-address> parameter displays the routes for a specified neighbor. You must specify this address in hexadecimal using 16-bit values between colons as documented in RFC 2373.

The **best** keyword displays the “best” routes, which are installed in the IPv6 route table.

The **unreachable** keyword displays the routes whose destinations are unreachable using any of the BGP4+ paths in the IPv6 route table.

The **detail** keyword displays detailed information about the routes. If you do not specify this parameter, a summary of the routes displays.

This display shows the following information:

**Table 8.19: Summary of best and unreachable routes from a BGP4+ neighbor**

| This Field... | Displays... |
|---|---|
| Number of accepted routes from a specified neighbor | The number of routes displayed by the command. |
| Status codes | A list of the characters the display uses to indicate the route’s status. The status code appears in the Status column of the display. The status codes are described in the command’s output. |
| Prefix | The route’s prefix. |
| Next Hop | The next-hop router for reaching the route from the router. |
| Metric | The value of the route’s MED attribute. If the route does not have a metric, this field is blank. |

**Table 8.19: Summary of best and unreachable routes from a BGP4+ neighbor (Continued)**

| This Field... | Displays... |
|---|---|
| LocPrf | The degree of preference for the route relative to other routes in the local AS. When the BGP4+ algorithm compares routes on the basis of local preferences, the route with the higher local preference is chosen. The preference can have a value from 0 – 4294967295. |
| Weight | The value that this router associates with routes from a specific neighbor. For example, if the router receives routes to the same destination from two BGP4+ neighbors, the router prefers the route from the neighbor with the larger weight. |
| Status | The route's status, which can be one or more of the following:<br>• A – AGGREGATE. The route is an aggregate route for multiple networks.<br>• B – BEST. BGP4+ has determined that this is the optimal route to the destination.<br>• C – CONFED_EBGP. The route was learned from a neighbor in the same confederation and AS, but in a different sub-AS within the confederation.<br>• D – DAMPED. This route has been dampened (by the route dampening feature), and is currently unusable.<br>• E – EBGP. The route was learned through a router in another AS.<br>• H – HISTORY. Route dampening is configured for this route, and the route has a history of flapping and is unreachable now.<br>• I – IBGP. The route was learned through a router in the same AS.<br>• L – LOCAL. The route originated on this router.<br>• M – MULTIPATH. BGP4+ load sharing is enabled and this route was selected as one of the best ones to the destination. The best route among the multiple paths also is marked with "B".<br>**Note**: If the "m" is shown in lowercase, the software was not able to install the route in the IPv6 route table.<br>• S – SUPPRESSED. This route was suppressed during aggregation and thus is not advertised to neighbors.<br>• F – FILTERED. This route was filtered out by BGP4+ route policies on the router, but the router saved updates containing the filtered routes. |
| AS-PATH | The AS-path information for the route. |

For example, to display detailed information about the best routes to a destination received from neighbor 2000:4::106, enter the following command:

```
BigIron# show ipv6 bgp neighbor 2000:4::106 routes detail best
       There are 2 accepted routes from neighbor 2000:4::106
Searching for matching routes, use ^C to quit...
Status A:AGGREGATE B:BEST b:NOT-INSTALLED-BEST C:CONFED_EBGP D:DAMPED
       E:EBGP H:HISTORY I:IBGP L:LOCAL M:MULTIPATH S:SUPPRESSED F:FILTERED
1       Prefix: 2002::/16,  Status: BE,  Age: 18h48m56s
         NEXT_HOP: 2000:4::106,  Learned from Peer: 2000:4::106 (65001)
          LOCAL_PREF: 100,  MED: 1,  ORIGIN: incomplete,  Weight: 0
           AS_PATH: 65001
2       Prefix: 2002:1234::/32,  Status: BE,  Age: 18h48m56s
         NEXT_HOP: 2000:4::106,  Learned from Peer: 2000:4::106 (65001)
          LOCAL_PREF: 100,  MED: 1,  ORIGIN: incomplete,  Weight: 0
           AS_PATH: 65001
```

This display shows the following information:

**Table 8.20: Detailed best and unreachable routes from a BGP4+ neighbor**

| This Field... | Displays... |
|---|---|
| Number of accepted routes from a specified neighbor (appears only in display for all routes) | For information about this field, see Table 8.19 on page 8-51. |
| Status codes | For information about this field, see Table 8.19 on page 8-51. |
| Prefix | For information about this field, see Table 8.19 on page 8-51. |
| Status | For information about this field, see Table 8.19 on page 8-51. |
| Age | The age of the route, in seconds. |
| Next Hop | For information about this field, see Table 8.19 on page 8-51. |
| Learned from Peer | The IPv6 address of the neighbor from which this route is learned. “Local Router” indicates that the router itself learned the route. |
| LOCAL_PREF | For information about this field, see Table 8.19 on page 8-51. |
| MED | The value of the RIB route’s MED attribute. If the route does not have a metric, this field is blank. |

**Table 8.20: Detailed best and unreachable routes from a BGP4+ neighbor (Continued)**

| This Field... | Displays... |
|---|---|
| Origin | The source of the route information. The origin can be one of the following:<br>• EGP – The routes with this set of attributes came to BGP4+ through EGP.<br>• IGP – The routes with this set of attributes came to BGP4+ through IGP.<br>• INCOMPLETE – The routes came from an origin other than one of the above. For example, they may have been redistributed from OSPFv3 or RIPng.<br>When BGP4+ compares multiple routes to a destination to select the best route, IGP is preferred over EGP and both are preferred over INCOMPLETE. |
| Weight | For information about this field, see Table 8.19 on page 8-51. |
| AS-PATH | For information about this field, see Table 8.19 on page 8-51. |

## Displaying IPv6 Neighbor Route Summary Information

You can display route summary information for all neighbors or a specified neighbor only.

For example, to display summary information for neighbor 2000:4::110, enter the following command at any level of the CLI:

```
BigIron# show ipv6 bgp neighbor 2000:4::110 routes-summary
1   IP Address: 2000:4::110
Routes Accepted/Installed:0,  Filtered/Kept:0,  Filtered:0
   Routes Selected as BEST Routes:0
      BEST Routes not Installed in IP Forwarding Table:0
   Unreachable Routes (no IGP Route for NEXTHOP):0
   History Routes:0

NLRIs Received in Update Message:0,  Withdraws:0 (0),  Replacements:0
   NLRIs Discarded due to
      Maximum Prefix Limit:0, AS Loop:0
      Invalid Nexthop:0, Invalid Nexthop Address:0.0.0.0
      Duplicated Originator_ID:0, Cluster_ID:0

Routes Advertised:2,  To be Sent:0,  To be Withdrawn:0
NLRIs Sent in Update Message:2,  Withdraws:0,  Replacements:0

Peer Out of Memory Count for:
   Receiving Update Messages:0, Accepting Routes(NLRI):0
   Attributes:0, Outbound Routes(RIB-out):0 Outbound Routes Holder:0
```

***Syntax:*** show ipv6 bgp neighbor [<ipv6-address>] routes-summary

This display shows the following information:

s

**Table 8.21: BGP4+ neighbor route summary information**

| This Field... | Displays... |
|---|---|
| IP Address | The IPv6 address of the neighbor |
| Routes Received | How many routes the router has received from the neighbor during the current BGP4+ session.<br>• Accepted/Installed – Indicates how many of the received routes the router accepted and installed in the BGP4+ route table.<br>• Filtered/Kept – Indicates how many routes were filtered out, but were nonetheless retained in memory for use by the soft reconfiguration feature.<br>• Filtered – Indicates how many of the received routes were filtered out. |
| Routes Selected as BEST Routes | The number of routes that the router selected as the best routes to their destinations. |
| BEST Routes not Installed in IPv6 Forwarding Table | The number of routes received from the neighbor that are the best BGP4+ routes to their destinations, but were nonetheless not installed in the IPv6 route table because the router received better routes from other sources (such as OSPFv3, RIPng, IPv6 IS-IS, or static IPv6 routes). |
| Unreachable Routes | The number of routes received from the neighbor that are unreachable because the router does not have a valid RIPng, OSPFv3, or static IPv6 route to the next hop. |
| History Routes | The number of routes that are down but are being retained for route flap dampening purposes. |
| NLRIs Received in Update Message | The number of routes received in Network Layer Reachability (NLRI) format in UPDATE messages.<br>• Withdraws – The number of withdrawn routes the router has received.<br>• Replacements – The number of replacement routes the router has received. |
| NLRIs Discarded due to | Indicates the number of times the router discarded an NLRI for the neighbor due to the following reasons:<br>• Maximum Prefix Limit – The router's configured maximum prefix amount had been reached.<br>• AS Loop – An AS loop occurred. An AS loop occurs when the BGP4+ AS-path attribute contains the local AS number.<br>• Invalid Nexthop Address – The next hop value was not acceptable.<br>• Duplicated Originator_ID – The originator ID was the same as the local router ID.<br>• Cluster_ID – The cluster list contained the local cluster ID, or contained the local router ID (see above) if the cluster ID is not configured. |

**Table 8.21: BGP4+ neighbor route summary information (Continued)**

| This Field... | Displays... |
|---|---|
| Routes Advertised | The number of routes the router has advertised to this neighbor.<br>• To be Sent – The number of routes the router has queued to send to this neighbor.<br>• To be Withdrawn – The number of NLRIs for withdrawing routes the router has queued up to send to this neighbor in UPDATE messages. |
| NLRIs Sent in Update Message | The number of NLRIs for new routes the router has sent to this neighbor in UPDATE messages.<br>• Withdraws – The number of routes the router has sent to the neighbor to withdraw.<br>• Replacements – The number of routes the router has sent to the neighbor to replace routes the neighbor already has. |
| Peer Out of Memory Count for | Statistics for the times the router has run out of BGP4+ memory for the neighbor during the current BGP4+ session.<br>• Receiving Update Messages – The number of times UPDATE messages were discarded because there was no memory for attribute entries.<br>• Accepting Routes(NLRI) – The number of NLRIs discarded because there was no memory for NLRI entries. This count is not included in the Receiving Update Messages count.<br>• Attributes – The number of times there was no memory for BGP4+ attribute entries.<br>• Outbound Routes (RIB-out) – The number of times there was no memory to place a “best” route into the neighbor's route information base (Adj-RIB-Out) for routes to be advertised.<br>• Outbound Routes Holder – For debugging purposes only. |

## Displaying BGP4+ Peer Group Configuration Information

You can display configuration information for all peer groups or a specified peer group configured on a router.

For example, to display configuration information for a peer group named peer1, enter the following command at any level of the CLI:

```
BigIron# show ipv6 bgp peer-group peer1
1   BGP peer-group is pg1, Remote AS: 65002
    Description: device group 1
       NextHopSelf: yes
      Address family : IPV4 Unicast
      Address family : IPV4 Multicast
      Address family : IPV6 Unicast
    Members:
       IP Address: 192.169.102.2
       IP Address: 192.169.100.2
       IP Address: 192.169.101.2
       IP Address: 192.169.103.2
       IP Address: 192.169.104.2
       IP Address: 192.169.105.2
       IP Address: 192.169.106.2
       IP Address: 192.169.107.2
       IP Address: 192.169.108.2
       IP Address: 192.169.109.2
       IP Address: 192.169.110.2
       IP Address: 192.169.111.2
       IP Address: 192.169.112.2
```

***Syntax:*** show ipv6 bgp peer-group [<peer-group-name>]

The display shows only parameters that have values different from their default settings.

## Displaying BGP4+ Summary

To view summary BGP4+ information for the router, enter the following command at any level of the CLI:

```
BigIron# show ipv6 bgp summary
BGP4 Summary
Router ID: 223.223.223.223   Local AS Number : 65001
Confederation Identifier : not configured
Confederation Peers:
Maximum Number of Paths Supported for Load Sharing : 1
Number of Neighbors Configured : 1
Number of Routes Installed : 2
Number of Routes Advertising to All Neighbors : 2
Number of Attribute Entries Installed : 1
Neighbor Address  AS#   State   Time     Rt:Accepted Filtered Sent   ToSend
2000:4::110       65002 ESTAB  21h32m32s    0        0        2      0
```

***Syntax:*** show ipv6 bgp summary

This display shows the following information:

**Table 8.22: BGP4+ summary information**

| This Field... | Displays... |
|---|---|
| Router ID | The router's router ID. |
| Local AS Number | The BGP4+ AS number in which the router resides. |
| Confederation Identifier | The AS number of the confederation in which the router resides. |
| Confederation Peers | The numbers of the local ASs contained in the confederation. This list matches the confederation peer list you configure on the router. |
| Maximum Number of Paths Supported for Load Sharing | The maximum number of route paths across which the router can balance traffic to the same destination. The feature is enabled by default but the default number of paths is 1. You can increase the number from 2 – 8 paths. For more information, see the "Configuring BGP4" chapter in the *Foundry Router Configuration Guide*. |
| Number of Neighbors Configured | The number of BGP4+ neighbors configured on this router. |
| Number of Routes Installed | The number of BGP4+ routes in the router's BGP4+ route table.<br>To display the BGP4+ route table, see "Displaying the BGP4+ Route Table" on page 8-13. |
| Number of Routes Advertising to All Neighbors | The total of the RtSent and RtToSend columns for all neighbors. |
| Number of Attribute Entries Installed | The number of BGP4+ route-attribute entries in the router's route-attributes table. To display the route-attribute table, see "Displaying BGP4+ Route-Attribute Entries" on page 8-20. |
| Neighbor Address | The IPv6 addresses of this router's BGP4+ neighbors. |
| AS# | The AS number. |

**Table 8.22: BGP4+ summary information (Continued)**

| This Field... | Displays... |
|---|---|
| State | The state of this router's neighbor session with each neighbor. The states are from this router's perspective of the session, not the neighbor's perspective. The state values can be one of the following for each router:<br>• IDLE – The BGP4+ process is waiting to be started. Usually, enabling BGP4+ or establishing a neighbor session starts the BGP4+ process.<br>  • A minus sign (-) indicates that the session has gone down and the software is clearing or removing routes.<br>• ADMND – The neighbor has been administratively shut down.<br>  • A minus sign (-) indicates that the session has gone down and the software is clearing or removing routes.<br>• CONNECT – BGP4+ is waiting for the connection process for the TCP neighbor session to be completed.<br>• ACTIVE – BGP4+ is waiting for a TCP connection from the neighbor.<br>**Note**: If the state frequently changes between CONNECT and ACTIVE, there may be a problem with the TCP connection.<br>• OPEN SENT – BGP4+ is waiting for an Open message from the neighbor.<br>• OPEN CONFIRM – BGP4+ has received an OPEN message from the neighbor and is now waiting for either a KEEPALIVE or NOTIFICATION message. If the router receives a KEEPALIVE message from the neighbor, the state changes to Established. If the message is a NOTIFICATION, the state changes to Idle.<br>• ESTABLISHED – BGP4+ is ready to exchange UPDATE packets with the neighbor.<br>  • If there is more BGP data in the TCP receiver queue, a plus sign (+) is also displayed.<br>**Note**: If you display information for the neighbor using the **show ipv6 bgp neighbor** <ipv6-address> command, the TCP receiver queue value will be greater than 0. |
| Time | The time that has passed since the state last changed. |
| Accepted | The number of routes received from the neighbor that this router installed in the BGP4+ route table. Usually, this number is lower than the RoutesRcvd number. The difference indicates that this router filtered out some of the routes received in the UPDATE messages. |
| Filtered | The routes or prefixes that have been filtered out.<br>• If soft reconfiguration is enabled, this field shows how many routes were filtered out (not placed in the BGP4+ route table) but retained in memory.<br>• If soft reconfiguration is not enabled, this field shows the number of BGP4+ routes that have been filtered out. |

**Table 8.22: BGP4+ summary information (Continued)**

| This Field... | Displays... |
|---|---|
| Sent | The number of BGP4+ routes that the router has sent to the neighbor. |
| ToSend | The number of routes the router has queued to send to this neighbor. |

# Chapter 9
# Configuring IPv4-to-IPv6 Transition Mechanisms

One strategy for transitioning an existing IPv4 topology to IPv6 is to deploy IPv6 in isolated domains while maintaining an IPv4 infrastructure. This strategy allows you to deploy IPv6 incrementally with minimal disruption to the IPv4 infrastructure. To support this strategy, Foundry in turn supports the following transition mechanisms:

- Dual stack backbone.
- IPv6 over IPv4 tunnels.

This chapter describes these two transition mechanisms and explains how to configure IPv6 over IPv4 tunnels, clear IPv6 tunnel statistics, and display IPv6 tunnel information.

## Dual Stack Backbone

***Dual stack backbone*** is a IPv4-to-IPv6 transition strategy that requires backbone routers and end systems to run both IPv4 and IPv6 protocol stacks. If you implement a dual stack backbone, you should be able to enable IPv4 and IPv6 routing protocols and all other features, as you would if you were running only one protocol stack, without any limitations. However, before implementing this strategy, you must consider the following:

- You must make certain that each backbone router has enough memory to handle IPv4 and IPv6 forwarding processes, routing protocols, and route tables.
- You must define and maintain IPv4 and IPv6 address schemes in your topology.
- You must configure, maintain, and manage IPv4 and IPv6 routing protocols.

This section provides information about the following topics:

- End system dual stack operation.
- Backbone router dual stack operation.

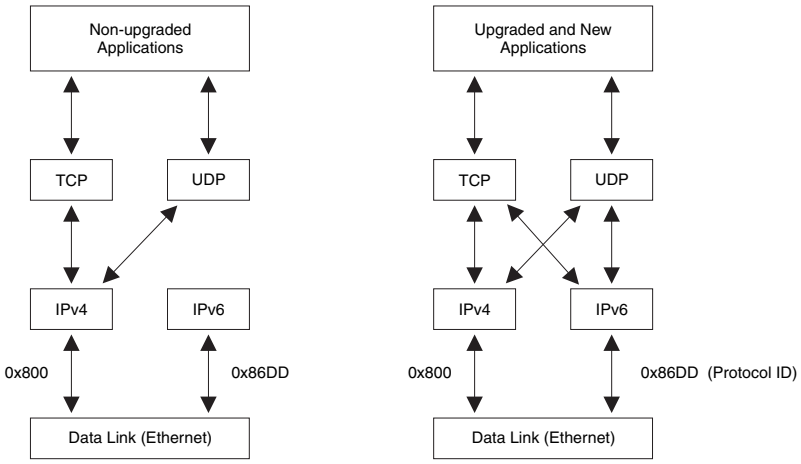
### End System Dual Stack Operation

When you configure an end system to run dual stacks, the following categories of applications can coexist on the end system:

- Existing applications that support the IPv4 stack and are not upgraded to support the IPv6 stack.
- Existing applications that support the IPv4 stack and are upgraded to support the IPv6 stack.
- Newly installed applications that support the IPv4 and IPv6 stacks.

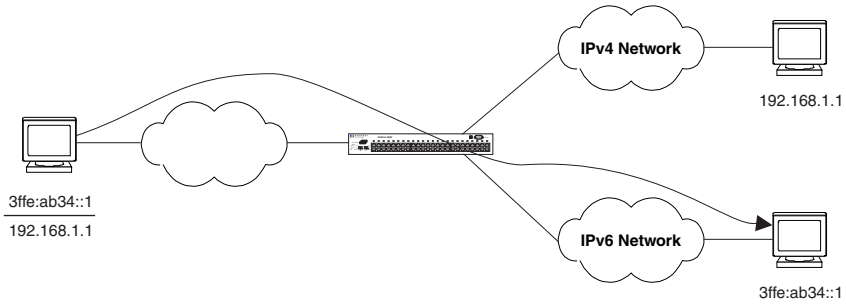
Figure 9.1 shows how the various applications on an end system function with dual stacks. Non-upgraded applications use the IPv4 stack only, while upgraded and new applications use both IPv4 and IPv6 stacks.

**Figure 9.1IPv4 and IPv6 dual stacks on an end system**




If an application supports both IPv4 and IPv6 stacks, the application decides which address, and thereby, which protocol to use. Generally, IPv6 is chosen by default. After the application decides on the address, it connects the source node to the destination using the chosen address. In Figure 9.2, the application decides to use the IPv6 address.

**Figure 9.2End system dual stack operation**




Some applications that support both IPv4 and IPv6 stacks, such as Telnet, SSH, and ping, allow you to explicitly specify either an IPv4 or IPv6 address. For more information about these applications, see “Managing a Foundry Device Over IPv6” on page 13-1.

## Backbone Router Dual Stack Operation

To implement a dual stack backbone, you must configure each backbone router to run both IPv4 and IPv6 protocol stacks. When implemented, IPv4 communication occurs using the IPv4 protocol stack and IPv4 packets are routed using IPv4 routing protocols. Likewise, IPv6 communication occurs using the IPv6 protocol stack, and IPv6 packets are routed using IPv6 routing protocols.

In other words, IPv4 and IPv6 forwarding processes, routing protocols, and route tables run in parallel to one another, which also means that they are separate and independent of one another. For example, you cannot redistribute the routes learned by an IPv4 routing protocol into an IPv6 routing protocol, or vice versa. The only situation in which the protocols intermingle is when tunneling IPv6 traffic over IPv4 tunnels (IPv6 packets are encapsulated within an IPv4 packet). For more information, see “IPv6 Over IPv4 Tunnels” next.

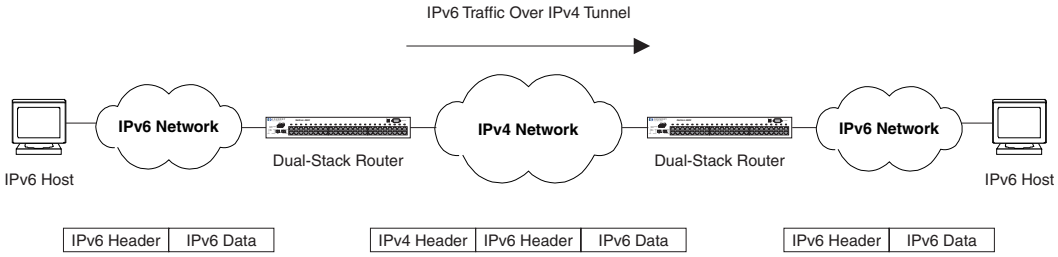
# IPv6 Over IPv4 Tunnels

**NOTE:** On the BigIron MG8 and NetIron 40G, this feature is available in software release 02.2.01 and later.

To enable communication between the isolated IPv6 domains using the IPv4 infrastructure, you can configure IPv6 over IPv4 tunnels. As shown in Figure 9.3, these tunnels encapsulate an IPv6 packet within an IPv4 packet.

**Figure 9.3IPv6 over an IPv4 tunnel**




Foundry supports the following IPv6 over IPv4 tunneling mechanisms:

- Manually configured tunnels
- Automatic 6to4 tunnels
- Automatic IPv4-compatible IPv6 tunnels

In general, a manually configured tunnel establishes a permanent link between routers in IPv6 domains, while the automatic tunnels establish a transient link that is created and taken down on an as-needed basis. (Although the feature name and description may imply otherwise, some configuration is necessary to set up an automatic tunnel.) Also, a manually configured tunnel has explicitly configured IPv4 addresses for the tunnel source and destination, while the automatic tunnels have an explicitly configured IPv4 address for the tunnel source and an automatically generated address for the tunnel destination.

---

**NOTE:** Foundry's implementation of IPv6 supports automatic IPv4-compatible IPv6 tunnels. However, because of this tunneling mechanism's inherent dependence on IPv4 addresses, which diminishes the benefits of IPv6, Foundry recommends using either manually configured tunnels or automatic 6to4 tunnels instead.

---

These tunneling mechanisms require that the router at each end of the tunnel run both IPv4 and IPv6 protocol stacks. (For information about configuring IPv4 and IPv6 protocol stacks on a router interface, see "Configuring IPv4 and IPv6 Protocol Stacks" on page 3-6.) The routers running both protocol stacks, or ***dual-stack routers***, can interoperate directly with both IPv4 and IPv6 end systems and routers.

## Configuring a Manual IPv6 Tunnel

You can use a manually configured tunnel to connect two isolated IPv6 domains. You should deploy this point-to-point tunnel mechanism if you need a permanent and stable connection.

To configure a manual IPv6 tunnel, enter commands such as the following on a Layer 3 Switch running both IPv4 and IPv6 protocol stacks on each end of the tunnel:

```
BigIron(config)# interface tunnel 1
BigIron(config-tnif-1)#ipv6 address 2001:b78:384d:34::/64 eui-64
BigIron(config-tnif-1)#tunnel source ethernet 3/1
BigIron(config-tnif-1)#tunnel destination 198.162.100.1
BigIron(config-tnif-1)#tunnel mode ipv6ip
```

This example creates tunnel interface 1 and assigns a global IPv6 address with an automatically computed EUI-64 interface ID to it. The IPv4 address assigned to Ethernet interface 3/1 is used as the tunnel source, while the IPv4 address 192.168.100.1 is configured as the tunnel destination. Finally, the tunnel mode is specified as a manual IPv6 tunnel.

***Syntax:*** interface tunnel <number>

For the <number> parameter, specify a value between 1 – 32.

***Syntax:*** ipv6 address <ipv6-prefix>/<prefix-length> [eui-64]

You must specify the <ipv6-prefix> parameter in hexadecimal using 16-bit values between colons as documented in RFC 2373.

You must specify the <prefix-length> parameter as a decimal value. A slash mark (/) must follow the <ipv6-prefix> parameter and keyword configures the global or site-local address with an EUI-64 interface ID in the low-order 64 bits. The interface ID is automatically constructed in IEEE EUI-64 format using the interface's MAC address.

***Syntax:*** tunnel source <ipv4-address> | ethernet <port> | loopback <number> | tunnel <number> | ve <number>

You must specify the <ipv4-address> parameter using 8-bit values in dotted decimal notation.

The **ethernet | loopback | tunnel | ve** parameter specifies an interface as the tunnel source. If you specify an Ethernet interface, also specify the port number associated with the interface. If you specify a loopback, VE, or tunnel interface, also specify the loopback, VE, or tunnel number, respectively.

***Syntax:*** tunnel destination <ipv4-address>

You must specify the <ipv4-address> parameter using 8-bit values in dotted decimal notation.

***Syntax:*** tunnel mode ipv6ip

## Configuring an Automatic 6to4 Tunnel

An automatic 6to4 tunnel establishes a transient link between IPv6 domains, which are connected by an IPv4 backbone. When needed, a device on which an automatic 6to4 tunnel is configured in one domain can establish a tunnel with another similarly configured device in another domain. When no longer needed, the devices take down the tunnel.

Instead of a manually configured tunnel destination, an automatic 6to4 tunnel constructs a globally unique 6to4 prefix, which determines the tunnel destination. The 6to4 prefix has the following format:

2002:<ipv4-address>::/48

For example, if the IP address is 12.1.1.2/24 the 6to4 prefix is 2002:0c01:0102::/48.

When two domains need to communicate, a device creates a tunnel using the 6to4 prefix. The software automatically generates the 6to4 prefix by concatenating a configured static IPv6 prefix of 2002 with the destination device's globally unique IPv4 address. (Each device in an IPv6 domain that needs to communicate over an automatic 6to4 tunnel must have one globally unique IPv4 address, from which the globally unique 6to4 prefix is constructed.) After the communication ends, the tunnel is taken down.

To configure an automatic 6to4 tunnel, enter commands such as the following on a device interface on each end of the tunnel. The devices at each end of the tunnel must run the IPv4 and IPv6 protocol stacks.

```
BigIron(config)# interface tunnel 1
BigIron(config-tnif-1)# ipv6 address 2002:0c01:0102::/64 eui-64
BigIron(config-tnif-1)# tunnel source ethernet 3/1
BigIron(config-tnif-1)# tunnel mode ipv6ip 6to4
BigIron(config-tnif-1)# exit
BigIron(config)# ipv6 route 2002::/16 tunnel 1
BigIron(config)# ipv6 route ::/0 tunnel 1
BigIron(config)# ipv6 route ::/0 tunnel 1 fe80::ao1:101
```

This example creates tunnel interface 1 and assigns a global IPv6 address with an automatically computed EUI-64 interface ID to it. If the IP address assigned to interface 3/1 is 12.1.1.2/24, then the IPv6 address for the tunnel interface is 2002:0c01:0102::/64. A static IPv6 prefix (2002::/16), which is used in the construction of a globally unique 6to4 prefix, is configured for the tunnel interface. The link local address of the tunnel interface is "fe80::ao1:101". Finally, the tunnel mode is specified as an IPv6 tunnel using a 6to4 prefix.

***Syntax:*** interface tunnel <number>

For the <number> parameter, specify a value between 1 – 32.

***Syntax:*** ipv6 address <ipv6-prefix>/<prefix-length> [eui-64]

You must specify the <ipv6-prefix> parameter in hexadecimal using 16-bit values between colons as documented in RFC 2373.

You must specify the <prefix-length> parameter as a decimal value. A slash mark (/) must follow the <ipv6-prefix> parameter and precede the <prefix-length> parameter.

The **eui-64** keyword configures the global or site-local address with an EUI-64 interface ID in the low-order 64 bits. The interface ID is automatically constructed in IEEE EUI-64 format using the interface's MAC address.

***Syntax:*** tunnel source <ipv4-address> | ethernet <port> | loopback <number> | tunnel <number> | ve <number>

You must specify the <ipv4-address> parameter using 8-bit values in dotted decimal notation.

The ethernet | loopback | tunnel | ve parameter specifies an interface as a tunnel source. If you specify an Ethernet interface, also specify the port number associated with the interface. If you specify a loopback, VE, or tunnel interface, also specify the loopback, VE, or tunnel number, respectively.

***Syntax:*** tunnel mode ipv6ip 6to4

***Syntax:*** ipv6 route 2002::/16 tunnel <number>

For the <number> parameter, you must specify the same tunnel number, which was created using the **interface tunnel** <number> command.

## Configuring an Automatic IPv4-Compatible IPv6 Tunnel

An IPv4-compatible IPv6 tunnel establishes a transient link between two IPv6 domains over an IPv4 backbone. When needed, a device on which an IPv4-compatible IPv6 tunnel is configured in one domain can establish a tunnel with another similarly configured device in another domain. When no longer needed, the devices take down the tunnel.

Instead of a manually configured tunnel destination, this tunnel constructs an IPv4-compatible IPv6 address, which determines the tunnel destination. When two domains need to communicate, a router creates a tunnel using the IPv4-compatible IPv6 address. The software automatically generates the IPv6 address by concatenating zeros in the high-order 96 bits and the device's globally unique IPv4 address in the low-order 32 bits (for example, 0:0:0:0:0:0:192.168.100.1). (Each router in an IPv6 domain that needs to communicate over an automatic IPv4-compatible IPv6 tunnel must have one globally unique IPv4 address, from which the globally unique IPv4-compatible IPv6 address is constructed.) After the communication ends, the tunnel is taken down.

**NOTE:** Foundry's implementation of IPv6 supports automatic IPv4-compatible IPv6 tunnels. However, because of this tunneling mechanism's inherent dependence on IPv4 addresses, which diminishes the benefits of IPv6, Foundry recommends using either manually configured tunnels or automatic 6to4 tunnels instead.

To configure an automatic IPv4-compatible IPv6 tunnel, enter commands such as the following on the device at each end of the tunnel. The devices at each end of the tunnel must run IPv4 and IPv6 protocol stacks.

```
BigIron(config)# interface tunnel 1
BigIron(config-tnif-1)#tunnel source ethernet 3/1
BigIron(config-tnif-1)#tunnel mode ipv6ip auto-tunnel
BigIron(config-tnif-1)#exit
BigIron(config)#ipv6 route ::/0 ::192.168.100.2
```

This example creates tunnel interface 1. The IPv4 address assigned to Ethernet interface 3/1 is used as the tunnel source. Specifying the tunnel mode as **ipv6ip auto-tunnel** sets up an IPv4-compatible tunnel using an IPv4-compatible IPv6 address.

***Syntax:*** interface tunnel <number>

For the <number> parameter, specify a value between 1 – 32.

***Syntax:*** tunnel source <ipv4-address> | ethernet <port> | loopback <number> | tunnel <number> | ve <number>

You must specify the <ipv4-address> parameter using 8-bit values in dotted decimal notation.

The **ethernet | loopback | tunnel | ve** parameter specifies an interface as the tunnel source. If you specify an Ethernet interface, also specify the port number associated with the interface. If you specify a loopback, VE, or tunnel interface, also specify the loopback, VE, or tunnel number, respectively.

***Syntax:*** tunnel mode ipv6ip auto-tunnel

## Clearing IPv6 Tunnel Statistics

You can clear all IPv6 tunnel statistics (reset all fields to zero) or statistics for a specified tunnel interface.

For example, to clear statistics for tunnel 1, enter the following command at the Privileged EXEC level or any of the Config levels of the CLI:

```
BigIron# clear ipv6 tunnel 1
```

***Syntax:*** clear ipv6 tunnel <number>

The <number> parameter specifies the tunnel number.

## Displaying IPv6 Tunnel Information

To display a summary of tunnel information, enter the following command at any level of the CLI:

```
BigIron# show ipv6 tunnel
IP6 Tunnels
  Tunnel  Mode         Packet Received  Packet Sent
  1       configured   0                0
  2       configured   0                22419
  6        6to4          0                  0
```

***Syntax:*** show ipv6 tunnel

This display shows the following information.

**Table 9.1: IPv6 tunnel information**

| This Field... | Displays... |
|---|---|
| Tunnel | The tunnel interface number. |
| Mode | The tunnel mode. Possible modes include the following:<br>• configured – Indicates a manually configured tunnel.<br>• 6to4 – Indicates an automatic 6to4 tunnel.<br>• auto – Indicates an automatic IPv4-compatible tunnel. |
| Packet Received | The number of packets received by a tunnel interface. |
| Packet Sent | The number of packets sent by a tunnel interface. |

## Displaying Tunnel Interface Information

For example, to display status and configuration information for tunnel interface 1, enter the following command at any level of the CLI:

```
BigIron# show interfaces tunnel 1
Tunnel1 is up, line protocol is up
  Hardware is Tunnel
  Tunnel source ethernet 3/5
  Tunnel destination is not configured
  Tunnel mode ipv6ip auto-tunnel
  No port name
  MTU 1500 bytes
```

***Syntax:*** show interfaces tunnel <number>

The <number> parameter indicates the tunnel interface number for which you want to display information.

This display shows the following information.

**Table 9.2: IPv6 tunnel interface information**

| This Field... | Displays... |
|---|---|
| Tunnel interface status | The status of the tunnel interface can be one of the following:<br>• up – The tunnel interface is functioning properly.<br>• down – The tunnel interface is not functioning and is down. |
| Line protocol status | The status of the line protocol can be one of the following:<br>• up – The line protocol is functioning properly.<br>• down – The line protocol is not functioning and is down. |
| Hardware is tunnel | The interface is a tunnel interface. |
| Tunnel source | The tunnel source can be one of the following:<br>• An IPv4 address<br>• The IPv4 address associated with an interface/port. |
| Tunnel destination | The tunnel destination can an IPv4 address. |
| Tunnel mode | The tunnel mode can be one the following:<br>• ipv6ip auto-tunnel – Indicates an automatic IPv4-compatible tunnel.<br>• ipv6ip 6to4 – Indicates an automatic 6to4 tunnel. |
| Port name | The port name configured for the tunnel interface. |
| MTU | The setting of the IPv6 maximum transmission unit (MTU). |

## Displaying Interface Level IPv6 Settings

To display Interface level IPv6 settings for tunnel interface 1, enter the following command at any level of the CLI:

```
BigIron MG8#show ipv6 inter tunnel 1
Interface Tunnel 1 is up, line protocol is up
  IPv6 is enabled, link-local address is fe80::3:4:2 [Preferred]
```

```
  Global unicast address(es):
    1001::1 [Preferred],  subnet is 1001::/64
    1011::1 [Preferred],  subnet is 1011::/64
  Joined group address(es):
    ff02::1:ff04:2
    ff02::5
    ff02::1:ff00:1
    ff02::2
    ff02::1
  MTU is 1480 bytes
  ICMP redirects are enabled
  No Inbound Access List Set
  No Outbound Access List Set
  OSPF enabled
```

The display command above reflects the following configuration:

BigIron MG8@M1#show running-config interface tunnel 1

```
!
interface tunnel 1
 port-name ManualTunnel1
 tunnel mode ipv6ip
 tunnel source loopback 1
 tunnel destination 2.1.1.1
 ipv6 address fe80::3:4:2 link-local
 ipv6 address 1011::1/64
 ipv6 address 1001::1/64
 ipv6 ospf area 0
```

# Chapter 10
# Configuring an IPv6 Access Control List

Foundry supports IPv6 access control lists (ACLs), which you can use for traffic filtering. You can configure up to 100 IPv6 ACLs.

An IPv6 ACL is composed of one or more conditional statements that pose an action (permit or deny) if a packet matches a specified source or destination prefix. There can be up to 1024 statements per device.

In ACLs with multiple statements, you can specify a priority for each statement.The specified priority determines the order in which the statement appears in the ACL.The last statement in each IPv6 ACL is an implicit deny statement for all packets that do not match the previous statements in the ACL.

You can configure an IPv6 ACL on a global basis, then apply it to the incoming or outgoing IPv6 packets on specified router interfaces. You can apply only one IPv6 ACL to an interface's incoming traffic and only one IPv6 ACL to an interface's outgoing traffic. When a router interface sends or receives an IPv6 packet, it applies the statement(s) within the ACL in their order of appearance to the packet. As soon as a match occurs, the router takes the specified action (permit or deny the packet) and stops further comparison for that packet.

**NOTE:** In release 02.0.02 for the NetIron IMR 640, IPv6 ACLs are supported on inbound and outbound traffic and are implemented in hardware, making it possible for the NetIron IMR 640 to filter traffic at line-rate speed on 10 Gigabit interfaces.

**NOTE:** In the IPv6 Release 01.0.00 for the NetIron 4802, only standard ACLs were supported, which filtered traffic based on source and destination IPv6 address. IPv6 release 02.0.00 replaced standard ACLs with extended filtering capabilities. IPv6 release 02.0.00 is backwards compatible with 01.0.00. When the device is upgraded to 02.0.00, ACLs in devices running release 01.0.00 will be converted to 02.0.00 syntax.

**NOTE:** On BigIron MG8 and NetIron 40G devices, ACLs are forwarded on hardware.

Foundry's IPv6 ACLs enable traffic filtering based on the following information:

- IPv6 protocol
- Source IPv6 address
- Destination IPv6 address
- IPv6 message type
- Source TCP or UDP port (if the IPv6 protocol is TCP or UDP)
- Destination TCP or UDP port (if the IPv6 protocol is TCP or UDP)

The IPv6 protocol can be one of the following well-known names or any IPv6 protocol number from 0 – 255:

- Authentication Header (AHP)
- Encapsulating Security Payload (ESP)
- Internet Control Message Protocol (ICMP)
- Internet Protocol Version 6 (IPv6)
- Stream Control Transmission Protocol (SCTP)
- Transmission Control Protocol (TCP)
- User Datagram Protocol (UDP)

For TCP and UDP, you also can specify a comparison operator and port name or number. For example, you can configure a policy to block web access to a specific website by denying all TCP port 80 (HTTP) packets from a specified source IPv6 address to the website's IPv6 address.

**NOTE:** In release 02.0.02 for the NetIron IMR 640, you can match packets for two TCP header flags using IPv6 ACLs. For syntax, see "For TCP" under "ACL Syntax".

IPv6 ACLs also provide support for filtering packets based on DSCP and flow label values.

This chapter contains the following sections:



## New Behavior for IPv6 ACLs (NetIron IMR 640 Release 03.0.00)

On the NetIron IMR 640, the behavior of IPv4 ACLs for dynamic trunk creation and deletion is that before a trunk is formed all ports which will be parts of the trunk must have the same configuration. For example, all of the ports can have no ACL, or have ACL 101 on inbound and outbound ports. After the trunk is removed, all ACL bindings (if there are any) are propagated to all of the secondary ports.

These rules were not previously applicable for IPv6 ACLs. With this release, they are now applicable for IPv6 ACLs.

## Using IPv6 ACLs as Input to Other Features

You can use an IPv6 ACL to provide input to other features such as route maps and distribution lists. When you use an ACL this way, use permit statements in the ACL to specify the traffic that you want to send to the other feature. If you use deny statements, the traffic specified by the deny statements is not supplied to the other feature.

## Configuring an IPv6 ACL

To configure an IPv6 ACL, you must do the following:

- Create the ACL.
- Apply the ACL to a router interface.

The following configuration tasks are optional:

- Control access to and from a router.

## Example Configurations

To configure an access list that blocks all Telnet traffic received on port 1/1 from IPv6 host 2000:2382:e0bb::2, enter the following commands.

```
BigIron(config)# ipv6 access-list fdry
BigIron(config-ipv6-access-list-fdry)# deny tcp host 2000:2382:e0bb::2 any eq
telnet
BigIron(config-ipv6-access-list-fdry)# permit ipv6 any any
BigIron(config-ipv6-access-list-fdry)# exit
BigIron(config)# int eth 1/1
BigIron(config-if-1/1)# ipv6 traffic-filter fdry in
BigIron(config)# write memory
```

Here is another example of commands for configuring an ACL and applying it to an interface.

```
BigIron(config)# ipv6 access-list netw
BigIron(config-ipv6-access-list-netw)# permit icmp 2000:2383:e0bb::/64
2001:3782::/64
BigIron(config-ipv6-access-list-netw)# deny ipv6 host 2000:2383:e0ac::2 host
2000:2383:e0aa:0::24
BigIron(config-ipv6-access-list-netw)# deny udp any any
BigIron(config-ipv6-access-list-netw)# permit ipv6 any any
```

The first condition permits ICMP traffic from hosts in the 2000:2383:e0bb::*x* network to hosts in the 2001:3782::*x* network.

The second condition denies all IPv6 traffic from host 2000:2383:e0ac::2 to host 2000:2383:e0aa:0::24.

The third condition denies all UDP traffic.

The fourth condition permits all packets that are not explicitly denied by the other entries. Without this entry, the ACL would deny all incoming or outgoing IPv6 traffic on the ports to which you assigned the ACL.

The following commands apply the ACL "netw" to the incoming and outgoing traffic on port 1/2 and to the incoming traffic on port 4/3.

```
BigIron(config)# int eth 1/2
BigIron(config-if-1/2)# ipv6 traffic-filter netw in
BigIron(config-if-1/2)# ipv6 traffic-filter netw out
BigIron(config-if-1/2)# exit
BigIron(config)# int eth 4/3
BigIron(config-if-4/3)# ipv6 traffic-filter netw in
BigIron(config)# write memory
```

Here is another example of an ACL:

```
BigIron(config)# ipv6 access-list nextone
BigIron(config-ipv6-access-list rtr)# deny tcp 2001:1570:21::/24
2001:1570:22::/24
BigIron(config-ipv6-access-list rtr)# deny udp any range 5 6 2001:1570:22::/24
BigIron(config-ipv6-access-list rtr)# permit ipv6 any any
BigIron(config-ipv6-access-list rtr)# write memory
```

The first condition in this ACL denies TCP traffic from the 2001:1570:21::*x* network to the 2001:1570:22::*x* network.

The next condition denies UDP packets from any source with source UDP port in ranges 5 to 6 and whose destination is to the 2001:1570:22::/24 network.

The third condition permits all packets containing source and destination addresses that are not explicitly denied by the first two. Without this entry, the ACL would deny all incoming or outgoing IPv6 traffic on the ports to which you assign the ACL.

A **show running-config** command displays the following:

```
 BigIron(config)# show running-config
ipv6 access-list rtr
 deny tcp 2001:1570:21::/24 2001:1570:22::/24
 deny udp any range rje 6 2001:1570:22::/24
 permit ipv6 any any
```

A **show ipv6 access-list** command displays the following:

```
 BigIron(config)# sh ipv6 access-list rtr
ipv6 access-list rtr: 3 entries
 10: deny tcp 2001:1570:21::/24 2001:1570:22::/24
 20: deny udp any range rje 6 2001:1570:22::/24
 30: permit ipv6 any any
```

The following commands apply the ACL “rtr” to the incoming traffic on ports 2/1 and 2/2.

```
BigIron(config)# int eth 2/1
BigIron(config-if-2/1)# ipv6 traffic-filter rtr in
BigIron(config-if-2/1)# exit
BigIron(config)# int eth 2/2
BigIron(config-if-2/2)# ipv6 traffic-filter rtr in
BigIron(config)# write memory
```

## Default and Implicit IPv6 ACL Action

The default action when no IPv6 ACLs are configured on the router is to permit all IPv6 traffic. However, once you configure an IPv6 ACL and apply it to a router interface, the default action for that interface is to deny all IPv6 traffic that is not explicitly permitted on the interface.

- If you want to tightly control access, configure ACLs consisting of permit entries for the access you want to permit. The ACLs implicitly deny all other access.
- If you want to secure access in environments with many users, you might want to configure ACLs that consist of explicit deny entries, then add an entry to permit all access to the end of each ACL. The router permits packets that are not denied by the deny entries.

Every IPv6 ACL has the following implicit conditions as its last match conditions:

1. **permit icmp any any nd-na** – Allows ICMP neighbor discovery acknowledgement.
2. **permit icmp any any nd-ns** – Allows ICMP neighbor discovery solicitation.
3. **deny ipv6 any any** – Denies IPv6 traffic. You must enter a **permit ipv6 any any** as the last statement in the access-list if you want to permit IPv6 traffic that were not denied by the previous statements.

The conditions are applied in the order shown above, with deny ipv6 any any as the last condition applied.

For example, if you want to deny ICMP neighbor discovery acknowledgement, then permit any remaining IPv6 traffic, enter commands such as the following:

```
BigIron(config)# ipv6 access-list netw
BigIron(config-ipv6-access-list-netw)# permit icmp 2000:2383:e0bb::/64
2001:3782::/64
BigIron(config-ipv6-access-list-netw)# deny icmp any any nd-na
BigIron(config-ipv6-access-list-netw)# permit ipv6 any any
```

The first permit statement permits ICMP traffic from hosts in the 2000:2383:e0bb::*x* network to hosts in the 2001:3782::*x* network.

The deny statement denies ICMP neighbor discovery acknowledgement.

The last entry permits all packets that are not explicitly denied by the other entries. Without this entry, the ACL will deny all incoming or outgoing IPv6 traffic on the ports to which you assigned the ACL.

Furthermore, if you add the statement **deny icmp any any** in the access list, then all neighbor discovery messages will be denied. You must explicitly enter the **permit icmp any any nd-na** and **permit icmp any any nd-ns** statements just before the **deny icmp** statement if you want the ACLs to permit neighbor discovery as in the example below.

```
BigIron(config)# ipv6 access-list netw
BigIron(config-ipv6-access-list-netw)# permit icmp 2000:2383:e0bb::/64
2001:3782::/64
BigIron(config-ipv6-access-list-netw)# permit icmp any any nd-na
BigIron(config-ipv6-access-list-netw)# permit icmp any any nd-ns
BigIron(config-ipv6-access-list-netw)# deny icmp any any
BigIron(config-ipv6-access-list-netw)# permit ipv6 any any
```

## ACL Syntax

**NOTE:** On BigIron MG8 and NetIron 40G devices, the following ACL features are not supported:

- **ipv6-operator flow-label**
- **tcp-operator** (all values).
- **ipv6-operator fragments** when any protocol is specified. The option "fragments" can be specified only when "permit/deny ipv6" is specified. If you specify "tcp" or any other protocol instead of "ipv6" the keyword, "fragments" cannot be used.
- **ipv6-operator routing**when any protocol is specified. (Same limitation as for **ipv6-operator fragments)**

When creating ACLs, use the appropriate syntax below for the protocol you are filtering.

### For IPv6 and Supported Protocols Other than ICMP, TCP, or UDP

***Syntax:*** [no] ipv6 access-list <acl name>

***Syntax:*** permit | deny <protocol>
<ipv6-source-prefix/prefix-length> | any | host <source-ipv6_address>
<ipv6-destination-prefix/prefix-length> | any | host <ipv6-destination-address>
[ipv6-operator [<value>]]

The **ipv6 access-list** <acl name> parameter enables the IPv6 configuration level and defines the name of the IPv6 ACL. The <acl name> can contain up to 199 characters and numbers, but cannot begin with a number and cannot contain any spaces or quotation marks.

The **permit** keyword indicates that the ACL will permit (forward) packets that match a policy in the access list.

The **deny** keyword indicates that the ACL will deny (drop) packets that match a policy in the access list.

The <protocol> parameter indicates the type of IPv6 packet you are filtering. You can specify a well-known name for some protocols whose number is less than 255. For other protocols, you must enter the number. Enter "?" instead of a protocol to list the well-known names recognized by the CLI. IPv6 protocols include:

- **AHP** – Authentication Header
- **ESP** – Encapsulating Security Payload
- **IPv6** – Internet Protocol version 6
- **SCTP** – Stream Control Transmission Protocol

The <ipv6-source-prefix>/<prefix-length> and <ipv6-destination-prefix>/<prefix-length> parameters specify a source or destination prefix and prefix length that a packet must match for the specified action (deny or permit) to occur. You must specify the <ipv6-source-prefix> and <ipv6-destination-prefix> parameters in hexadecimal using 16-bit values between colons as documented in RFC 2373. You must specify the <prefix-length> parameter as a decimal value. A slash mark (/) must follow the <ipv6-prefix> parameter and precede the <prefix-length> parameter.

The **any** keyword, when specified instead of the <ipv6-source-prefix>/<prefix-length> or <ipv6-destination-prefix>/<prefix-length> parameters, matches any IPv6 prefix and is equivalent to the IPv6 prefix ::/0.

The **host** <ipv6-source-address> and **host** <ipv6-destination-address> parameter lets you specify a host IPv6 address. When you use this parameter, you do not need to specify the prefix length. A prefix length of all128 is implied.

The **ipv6-operator** [<value>] parameter allows you to filter the packets further by using one of the following options:

- **dscp** – The policy applies to packets that match the traffic class value in the traffic class field of the IPv6 packet header. This operator allows you to filter traffic based on TOS or IP precedence. You can specify a value from 0 – 63.
- **flow-label** – The policy applies to packets that match the flow label value in the flow label field of the IPv6 packet header. You can specify a value from 0 – 1048575.
- **fragments** – The policy applies to fragmented packets that contain a non-zero fragment offset.

  **NOTE:** This option is not applicable to filtering based on source or destination port, TCP flags, and ICMP flags.

- **routing** – The policy applies to only IPv6 source-routed packets.
- **sequence** – The sequence parameter specifies where the conditional statement is to be added in the access list. You can add a conditional statement at particular place in an access list by specifying the entry number using the sequence keyword. You can specify a value from 1 – 4294967295.

### For ICMP

***Syntax:*** [no] ipv6 access-list <acl name>

***Syntax:*** permit | deny icmp <ipv6-source-prefix/prefix-length> | any | host <source-ipv6_address> <ipv6-destination-prefix/prefix-length> | any | host <ipv6-destination-address>
[ipv6-operator [<value>]]
[ [<icmp-type>][<icmp-code>] ] | [<icmp-messge>]

The **ipv6 access-list** <acl name> parameter enables the IPv6 configuration level and defines the name of the IPv6 ACL. The <acl name> can contain up to 199 characters and numbers, but cannot begin with a number and cannot contain any spaces or quotation marks.

The **permit** keyword indicates that the ACL will permit (forward) packets that match a policy in the access list.

The **deny** keyword indicates that the ACL will deny (drop) packets that match a policy in the access list.

The **icmp** keyword indicates the you are filtering ICMP packets.

The <ipv6-source-prefix>/<prefix-length> and <ipv6-destination-prefix>/<prefix-length> parameters specify a source or destination prefix and prefix length that a packet must match for the specified action (deny or permit) to occur. You must specify the <ipv6-source-prefix> and <ipv6-destination-prefix> parameters in hexadecimal using 16-bit values between colons as documented in RFC 2373. You must specify the <prefix-length> parameter as a decimal value. A slash mark (/) must follow the <ipv6-prefix> parameter and precede the <prefix-length> parameter.

The **any** keyword, when specified instead of the <ipv6-source-prefix>/<prefix-length> or <ipv6-destination-prefix>/<prefix-length> parameters, matches any IPv6 prefix and is equivalent to the IPv6 prefix ::/0.

The **host** <ipv6-source-address> and **host** <ipv6-destination-address> parameter lets you specify a host IPv6 address. When you use this parameter, you do not need to specify the prefix length. A prefix length of all128 is implied.

The **ipv6-operator** [<value>] parameter allows you to filter the packets further by using one of the following options:

- **dscp** – The policy applies to packets that match the traffic class value in the traffic class field of the IPv6 packet header. You can specify a value from 0 – 63.
- **flow-label** – The policy applies to packets that match the flow label value in the flow label field of the IPv6 packet header. You can specify a value from 0 – 1048575.
- **fragments** – The policy applies to fragmented packets that contain a non-zero fragment offset.

  **NOTE:** This option is not applicable to filtering based on source or destination port, TCP flags, and ICMP flags.

- **routing** – The policy applies to only IPv6 source-routed packets.
- **sequence** – The sequence parameter specifies where the conditional statement is to be added in the access list. You can add a conditional statement at particular place in an access list by specifying the entry number using the sequence keyword. You can specify a value from 1 – 4294967295.
- **<cr>** – Enters the ACL entry without an IPv6 operator.

You can specify an ICMP type and ICMP code or an ICMP message type. To specify an ICMP type, enter a value between 0–255 for the <icmp-type> parameter. To specify an ICMP code, enter a value between 0–255 for the <icmp-code> parameter.

If you want to specify an ICMP message, you can enter one of the following:

- beyond-scope
- destination-unreachable
- dscp
- echo-reply
- echo-request
- flow-label
- fragments
- header
- hop-limit
- mld-query
- mld-reduction
- mld-report
- nd-na
- nd-ns

- next-header
- no-admin
- no-route
- packet-too-big
- parameter-option
- parameter-problem
- port-unreachable
- reassembly-timeout
- renum-command
- renum-result
- renum-seq-number
- router-advertisement
- router-renumbering
- router-solicitation
- routing
- sequence
- time-exceeded
- unreachable

**NOTE:** If you do not specify a message type, the ACL applies to all types ICMP messages types.

### For TCP

***Syntax:*** [no] ipv6 access-list <acl name>

***Syntax:*** permit | deny <tcp>
<ipv6-source-prefix/prefix-length> | any | host <source-ipv6_address> [tcp-udp-operator [source-port-number]]
<ipv6-destination-prefix/prefix-length> | any | host <ipv6-destination-address> [tcp-udp-operator [destination-port- number]]
[ipv6-operator [<value>]] [tcp-operator [<value>]]

In release 02.0.02 for the NetIron IMR 640, the syntax is as follows:

***Syntax:*** [no] ipv6 access-list <acl name>

***Syntax:*** permit | deny <tcp>
<source-ipv6-prefix/prefix-length> | any | host <source-ipv6_address>
<destination-ipv6-prefix/prefix-length> | any | host <destination-ipv6-address>
[<tcp-operator-value>]

The **ipv6 access-list** <acl name> parameter enables the IPv6 configuration level and defines the name of the IPv6 ACL. The <acl name> can contain up to 199 characters and numbers, but cannot begin with a number and cannot contain any spaces or quotation marks.

The **permit** keyword indicates that the ACL will permit (forward) packets that match a policy in the access list.

The **deny** keyword indicates that the ACL will deny (drop) packets that match a policy in the access list.

The <tcp> parameter indicates the you are filtering TCP packets.

The <ipv6-source-prefix>/<prefix-length> and <ipv6-destination-prefix>/<prefix-length> parameters specify a source or destination prefix and prefix length that a packet must match for the specified action (deny or permit) to occur. You must specify the <ipv6-source-prefix> and <ipv6-destination-prefix> parameters in hexadecimal using 16-bit values between colons as documented in RFC 2373. You must specify the <prefix-length> parameter as a decimal value. A slash mark (/) must follow the <ipv6-prefix> parameter and precede the <prefix-length> parameter.

The **any** keyword, when specified instead of the <ipv6-source-prefix>/<prefix-length> or <ipv6-destination-prefix>/<prefix-length> parameters, matches any IPv6 prefix and is equivalent to the IPv6 prefix ::/0.

The **host** <ipv6-source-address> and **host** <ipv6-destination-address> parameter lets you specify a host IPv6 address. When you use this parameter, you do not need to specify the prefix length. A prefix length of all128 is implied.

The <tcp-udp-operator> parameter can be one of the following:

- **eq** – The policy applies to the TCP or UDP port name or number you enter after **eq**.
- **gt** – The policy applies to TCP or UDP port numbers greater than the port number or the numeric equivalent of the port name you enter after **gt**. Enter "?" to list the port names.
- **lt** – The policy applies to TCP or UDP port numbers that are less than the port number or the numeric equivalent of the port name you enter after **lt**.
- **neq** – The policy applies to all TCP or UDP port numbers except the port number or port name you enter after **neq**.
- **range** – The policy applies to all TCP port numbers that are between the first TCP or UDP port name or number and the second one you enter following the **range** parameter. The range includes the port names or numbers you enter. For example, to apply the policy to all ports between and including 23 (Telnet) and 53 (DNS), enter the following: **range 23 53**. The first port number in the range must be lower than the last number in the range.

The <source-port number> and <destination-port-number> for the tcp-udp-operator is the number of the port.

The **ipv6-operator** [<value>] parameter allows you to filter the packets further by using one of the following options:

- dscp – The policy applies to packets that match the traffic class value in the traffic class field of the IPv6 packet header. You can specify a value from 0 – 63.
- **flow-label** – The policy applies to packets that match the flow label value in the flow label field of the IPv6 packet header. You can specify a value from 0 – 1048575.
- **fragments** – The policy applies to fragmented packets that contain a non-zero fragment offset.

  **NOTE:** This option is not applicable to filtering based on source or destination port, TCP flags, and ICMP flags.

- **routing** – The policy applies to only IPv6 source-routed packets.
- **sequence** – The sequence parameter specifies where the conditional statement is to be added in the access list. You can add a conditional statement at particular place in an access list by specifying the entry number using the sequence keyword. You can specify a value from 1 – 4294967295.

The **tcp-operator** [<value>] parameter specifies a comparison operator for the TCP port. This parameter applies only when you specify **tcp** as the IP protocol. You can enter one of the following operators:

- **ack** – The policy applies to TCP packets with the ACK (Acknowledgment) bits set on (set to “1”) in the Control Bits field of the TCP packet header.
- **established** – This operator applies only to TCP packets. If you use this operator, the policy applies to TCP packets that have the ACK (Acknowledgment) or RST (Reset) bits set on (set to “1”) in the Control Bits field of the TCP packet header. Thus, the policy applies only to established TCP sessions, not to new sessions.
- **fin** – The policy applies to TCP packets with the FIN (Finish) bits set on (set to “1”) in the Control Bits field of

the TCP packet header.

- **psh** – The policy applies to all TCP packets with the PSH (Push) bit set on (set to "1") in the Control Bits field of the TCP packet header.
- **rst** – The policy applies to TCP packets with the RST (Reset) bits set on (set to “1”) in the Control Bits field of the TCP packet header.
- **syn** – The policy applies to TCP packets with the SYN (Synchronize) bits set on (set to “1”) in the Control Bits field of the TCP packet header.
- **urg** – The policy applies to TCP packets with the URG (Urgent) bits set on (set to “1”) in the Control Bits field of the TCP packet header.

In release 02.0.02 for the NetIron IMR 640, the available values for the [<tcp-operator-value>] parameter are **established** and **syn**.

### For UDP

***Syntax:*** [no] ipv6 access-list <acl name>

***Syntax:*** permit | deny <udp>
<ipv6-source-prefix/prefix-length> | any | host <source-ipv6_address> [tcp-udp-operator [source port number]]
<ipv6-destination-prefix/prefix-length> | any | host <ipv6-destination-address> [tcp-udp-operator [destination port number]]
[ipv6-operator [<value>]]

The **ipv6 access-list** <acl name> parameter enables the IPv6 configuration level and defines the name of the IPv6 ACL. The <acl name> can contain up to 199 characters and numbers, but cannot begin with a number and cannot contain any spaces or quotation marks.

The **permit** keyword indicates that the ACL will permit (forward) packets that match a policy in the access list.

The **deny** keyword indicates that the ACL will deny (drop) packets that match a policy in the access list.

The <udp> parameter indicates the you are filtering UDP packets.

The <ipv6-source-prefix>/<prefix-length> and <ipv6-destination-prefix>/<prefix-length> parameters specify a source or destination prefix and prefix length that a packet must match for the specified action (deny or permit) to occur. You must specify the <ipv6-source-prefix> and <ipv6-destination-prefix> parameters in hexadecimal using 16-bit values between colons as documented in RFC 2373. You must specify the <prefix-length> parameter as a decimal value. A slash mark (/) must follow the <ipv6-prefix> parameter and precede the <prefix-length> parameter.

The **any** keyword, when specified instead of the <ipv6-source-prefix>/<prefix-length> or <ipv6-destination-prefix>/<prefix-length> parameters, matches any IPv6 prefix and is equivalent to the IPv6 prefix ::/0.

The **host** <ipv6-source-address> and **host** <ipv6-destination-address> parameter lets you specify a host IPv6 address. When you use this parameter, you do not need to specify the prefix length. A prefix length of all128 is implied.

The <tcp-udp-operator> parameter can be one of the following:

- **eq** – The policy applies to the TCP or UDP port name or number you enter after **eq**.
- **gt** – The policy applies to TCP or UDP port numbers greater than the port number or the numeric equivalent of the port name you enter after **gt**. Enter "?" to list the port names.
- **lt** – The policy applies to TCP or UDP port numbers that are less than the port number or the numeric equivalent of the port name you enter after **lt**.
- **neq** – The policy applies to all TCP or UDP port numbers except the port number or port name you enter after **neq**.
- **range** – The policy applies to all UDP port numbers that are between the first TCP or UDP port name or number and the second one you enter following the **range** parameter. The range includes the port names or numbers you enter. For example, to apply the policy to all ports between and including 23 (Telnet) and 53 (DNS), enter the following: **range 23 53**. The first port number in the range must be lower than the last

number in the range.

The <source-port number> and <destination-port-number> for the tcp-udp-operator is the number of the source and destination port.

The **ipv6-operator** [<value>] parameter allows you to filter the packets further by using one of the following options:

- **dscp** – The policy applies to packets that match the traffic class value in the traffic class field of the IPv6 packet header. You can specify a value from 0 – 63.
- **flow-label** – The policy applies to packets that match the flow label value in the flow label field of the IPv6 packet header. You can specify a value from 0 – 1048575.
- **fragments** – The policy applies to fragmented packets that contain a non-zero fragment offset.

  **NOTE:** This option is not applicable to filtering based on source or destination port, TCP flags, and ICMP flags.

- **routing** – The policy applies to only IPv6 source-routed packets.
- **sequence** – The sequence parameter specifies where the conditional statement is to be added in the access list. You can add a conditional statement at particular place in an access list by specifying the entry number using the sequence keyword. You can specify a value from 1 – 4294967295.

## Filtering Packets Based on Flow Label and DSCP Values

To filter packets based on flow label and DSCP values, enter commands such as the following:

```
BigIron(config)# ipv6 access-list netw
BigIron(config-ipv6-access-list netw) deny ipv6 any any dscp 3
BigIron(config-ipv6-access-list netw)# deny ipv6 any any flow-label 345
```

***Syntax:*** ipv6 access-list <name>
deny | permit
<ipv6-source-prefix>/<prefix-length> | any
<ipv6-destination-prefix>/<prefix-length> | any [sequence <number>]
dscp <dscp-value> | flow-label <flow-label-value>

The **deny** keyword specifies that the request is denied if it matches the specified source and destination prefixes.

The **permit** keyword specifies that the request is permitted if it matches the specified source and destination prefixes.

The <ipv6-source-prefix>/<prefix-length> and <ipv6-destination-prefix>/<prefix-length> parameters specify a source or destination prefix and prefix length that a packet must match for the specified action (deny or permit) to occur. You must specify the <ipv6-source-prefix> and <ipv6-destination-prefix> parameters in hexadecimal using 16-bit values between colons as documented in RFC 2373. You must specify the <prefix-length> parameter as a decimal value. A slash mark (/) must follow the <ipv6-prefix> parameter and precede the <prefix-length> parameter.

The **any** keyword, when specified instead of the <ipv6-source-prefix>/<prefix-length> or <ipv6-destination-prefix>/<prefix-length> parameters, matches any IPv6 prefix and is equivalent to the IPv6 prefix ::/0.

Enter a value from 0 - 63 for the **dscp** <dscp-value> parameter if you want to filter packets based on their DSCP value.

Enter a value from 0 -1048575 for the **flow-label** <flow-label-value> parameter if you want to filter packets based on their flow value.

# Applying an IPv6 ACL to a Router Interface

To apply an IPv6 ACL, for example “access1”, to a router interface, enter commands such as the following:

```
BigIron(config)# interface ethernet 3/1
BigIron(config-if-e100-3/1)# ipv6 traffic-filter access1 in
```

This example applies the IPv6 ACL “access1” to incoming IPv6 packets on Ethernet interface 3/1. As a result, Ethernet interface 3/1 denies all incoming packets from the site-local prefix fec0:0:0:2::/64 and the global prefix 2001:100:1::/48 and permits all other incoming packets.

***Syntax:*** ipv6 traffic-filter <ipv6-acl-name> in | out

For the <ipv6-acl-name> parameter, specify the name of an IPv6 ACL created using the **ipv6 access-list** command.

The **in** keyword applies the specified IPv6 ACL to incoming IPv6 packets on the router interface.

The **out** keyword applies the specified IPv6 ACL to outgoing IPv6 packets on the router interface.

## Controlling Access to a Router

You can use an IPv6 ACL to filter control incoming and outgoing connections to and from a router. To do so, you must create an ACL and then specify the sequence in which the ACL is applied to incoming or outgoing connections to the router.

For example, to permit incoming connections from remote hosts (2000:2383:e0bb::2/128 and 2000:2383:e0bb::3/128) to a router (30ff:3782::ff89/128), enter the following commands:

```
BigIron(config)# ipv6 access-list remote-hosts permit 2000:2383:e0bb::2/128
30ff:3782::ff89/128 sequence 10
BigIron(config)# ipv6 access-list remote-hosts permit 2000:2383:e0bb::3/128
30ff:3782::ff89/128 sequence 20
BigIron(config)# ipv6 access-class remote-hosts in
```

Because of the implicit deny command at the end of each IPv6 ACL, the router denies incoming connections from all other IPv6 hosts.

***Syntax:*** ipv6 access-list <name> deny | permit <ipv6-source-prefix>/<prefix-length> | any <ipv6-destination-prefix>/<prefix-length> | any [sequence <number>]

The <name> parameter specifies a name for the IPv6 ACL. An IPv6 ACL name cannot start with a numeral, for example, 1access. Also, an IPv4 ACL and an IPv6 ACL cannot share the same name.

The **deny** keyword specifies that the request from the remote host is denied if it matches the specified source and destination prefixes.

The **permit** keyword specifies that the request from the remote host is permitted if it matches the specified source and destination prefixes.

The <ipv6-source-prefix>/<prefix-length> and <ipv6-destination-prefix>/<prefix-length> parameters specify a source or destination prefix and prefix length that a request must match for the specified action (deny or permit) to occur. You must specify the <ipv6-source-prefix> or <ipv6-destination-prefix> parameter in hexadecimal using 16-bit values between colons as documented in RFC 2373. You must specify the <prefix-length> parameter as a decimal value. A slash mark (/) must follow the <ipv6-prefix> parameter and precede the <prefix-length> parameter.

The **any** parameter, when specified instead of the <ipv6-source-prefix> or <ipv6-destination-prefix> parameter, matches any IPv6 prefix and is equivalent to the IPv6 prefix ::/0.

The **sequence** <number> parameter specifies the order in which a statement appears in an IPv6 ACL and is therefore applied to a request. You can specify a value from 0 – 4294967295.

## Adding a Comment to an IPv6 ACL Entry

You can optionally add a comment to describe entries in an IPv6 ACL. The comment appears in the output of **show** commands that display ACL information.

You can add a comment by entering the **remark** command immediately preceding an ACL entry, or specify the ACL entry to which the comment applies.

For example, to enter comments for preceding an ACL entry, enter commands such as the following:

```
BigIron(config)#ipv6 access-list rtr
BigIron(config-ipv6-access-list rtr)# remark This entry permits ipv6 packets from
3002::2 to any destination
BigIron(config-ipv6-access-list rtr)# permit ipv6 host 3000::2 any
BigIron(config-ipv6-access-list rtr)# remark This entry denies udp packets from
any source to any destination
BigIron(config-ipv6-access-list rtr)# deny udp any any
BigIron(config-ipv6-access-list rtr)# remark This entry denies IPv6 packets from
any source to any destination
BigIron(config-ipv6-access-list rtr)# deny ipv6 any any
BigIron(config-ipv6-access-list rtr)# write memory
```

***Syntax:*** remark <comment-text>

The <comment-text> can be up to 256 characters in length.

To apply a comment to a specific ACL entry, specify the ACL's entry number with the **remark-entry sequence** command. Use the **show ipv6 access-list** command to list ACL entry number. Enter commands such as the following :

```
BigIron(config)# ipv6 access-list netw
BigIron(config-ipv6-access-list netw) remark-entry sequence 10 This entry permits
ipv6 packets from 3000::2 to any destination
BigIron(config-ipv6-access-list netw)# remark-entry sequence 20 This entry denies
UDP packets from any source to any destination
BigIron(config-ipv6-access-list netw)# remark-entry sequence 30 This entry denies
IPv6 packets from any source to any destination
```

***Syntax:*** remark-entry sequence <sequence number> <comment-text>

The <sequence number> is the line number assigned to the ACL entry. For a list of ACL entry numbers, use the **show ipv6 access-list** command.

The <comment-text> can be up to 256 characters in length. The comment must be entered separately from the actual ACL entry; that is, you cannot enter the ACL entry and the ACL comment with the same command.

You can use the **show running-config** or **show ipv6 access-list** commands to display IPv6 ACLs and comments.

The following shows the comment text for the ACL named "rtr" in a show running-config display:

```
BigIron# show running-config
ipv6 access-list rtr
 remark This entry permits ipv6 packets from 3002::2 to any destination
 permit ipv6 host 3000::2 any
 remark This entry denies udp packets from any source to any destination
 deny udp any any
 remark This entry denies IPv6 packets from any source to any destination
 deny ipv6 any any
```

***Syntax:*** show running-config

The following example shows the comment text for the ACL named "rtr" in a **show ipv6 access-list** display:

```
BigIron# show ipv6 access-list rtr
ipv6 access-list rtr: 3 entries
 10: remark This entry permits ipv6 packets from 3002::2 to any destination
 10: permit ipv6 host 3000::2 any
 20: remark This entry denies udp packets from any source to any destination
 20: deny udp any any
 30: remark This entry denies IPv6 packets from any source to any destination
 30: deny ipv6 any any
```

***Syntax:*** show ipv6 access-list [<access-list-name>]

For the <access-list-name> parameter, specify the name of an IPv6 ACL created using the **ipv6 access-list** command.

Use the **all** keyword to display all IPv6 ACLs configured on the device.

## Displaying ACLs

To display the ACLs configured on a device, enter the **show ipv6 access-list** command. Here is an example:

```
BigIron# show ipv6 access-list
ipv6 access-list fdry: 8 entries
 10: permit ipv6 host 3000::2 any
 20: permit udp 3000::/16 any gt nfs
 30: deny icmp host 5000::5 host 6000::3 echo-request
 40: permit ipv6 host 3002::2 any
 50: deny udp 3000::/16 4000::/16 gt nfs
 60: permit tcp any any established
 70: permit udp any any gt nfs
 80: remark this is last entryipv6 access-list fdry: 3 entries
```

***Syntax:*** show ipv6 access-list [<access-list-name>]

## Displaying Statistics for IPv6 ACL Accounting for the NetIron IMR 640

**NOTE:** This is available in release 02.0.02 of the NetIron IMR 640.

To display statistics for IPv6 accounting, enter commands such as the following:

```
NetIron IMR640 Router(config)#show ipv6 access-list accounting brief
Collecting IPv6 ACL accounting summary for 5/1 ...  Completed successfully.
Collecting IPv6 ACL accounting summary for 5/2 ...  Completed successfully.
IPv6 ACL Accounting Summary: (ac = accumulated since accounting started)
    Int    In ACL              Total In Hit   Out ACL             Total Out Hit
    5/1   fdry115               3551122(1s)
                              135155472(1m)
                                      0(5m)
                              135155472(ac)
    5/2                                      fdry116               3551123(1s)
                                                                 135154337(1m)
                                                                         0(5m)
                                                                 135154337(ac)
```

The display shows the following information:

| This Field... | Displays... |
|---|---|
| The IP multicast traffic snooping state | The first line of the display indicates whether IP multicast traffic snooping is enabled or disabled. If enabled, it indicates if the feature is configured as passive or active. |
| Collecting ACL accounting summary for <interface> | Shows for which interfaces the ACL accounting information was collected and whether or not the collection was successful. |
| Int | The ID of the interface for which the statistics are being reported. |
| In ACL | The ID of the ACL used to filter the incoming traffic on the interface. |
| Total In Hit* | The number of hits from incoming traffic processed by all ACL entries (filters) in the ACL. A number is shown for each counter. |
| Out ACL | ID of the ACL used to filter the outgoing traffic on the interface. |
| Total Out Hit* | The number of hits from incoming traffic processed by all ACL entries (filters) in the ACL. A number is shown for each counter. |
| * The Total In Hit and Total Out Hit displays the total number of hits for all the ACL entries (or filters) in an ACL. For example, if an ACL has five entries and each entry processed matching conditions three times during the last minute, then the total Hits for the 1m counter is 15. | |

***Syntax:*** show ipv6 access-list accounting brief

## Displaying IPv6 Accounting Statistics for an Interface on the NetIron IMR 640

**NOTE:** This is available in release 02.0.02 of the NetIron IMR 640.

To display statistics for an interface, enter commands such as the following:

```
NetIron IMR640 Router(config)#show ipv6 access-list accounting eth 5/1 in

Collecting IPv6 ACL accounting for 5/1  ...  Completed successfully.

IPv6 ACL Accounting Information:

Inbound: IPv6 ACL fdry115
   10:  permit ipv6 host 4000::2 any
       Hit count: (1 sec)              3551111   (1 min)            135155472
                  (5 min)                    0   (accum)            135155472
```

The display shows the following information:

| This Field... | Displays... |
|---|---|
| The IP multicast traffic snooping state | The first line of the display indicates whether IP multicast traffic snooping is enabled or disabled. If enabled, it indicates if the feature is configured as passive or active. |
| Collecting IPv6 ACL accounting for <interface> | Shows the interface included in the report and whether or not the collection was successful. |
| Outbound/Inbound ACL ID | Shows the direction of the traffic on the interface and the ID of the ACL used. |
| # | Shows the priority of the ACL entry, followed by the permit or deny condition defined for that ACL entry.<br>ACL entries are displayed in order of ascending ACL filter priorities. |
| Hit count | Shows the number of hits for each counter. |

***Syntax:*** show ipv6 access-list accounting ethernet [<slot>/<port> | ve <ve-number>]  in | out

Use **ethernet** <slot>/<port> to display a report for a physical interface.

Use **ve** <ve-number> to display a report for the ports that are included in a virtual routing interface. For example, if ports 1/2, 1/4, and 1/6 are all members of ve 2, the report includes information for all three ports.

Use the **in** parameter to display statistics for incoming traffic; **out** for outgoing traffic.

# Chapter 11
# Configuring an IPv6 Prefix List

This chapter describes the following:

- How to configure an IPv6 prefix list.
- How to display IPv6 prefix list information.

## Configuring an IPv6 Prefix List

Foundry supports IPv6 prefix lists, which you can use for basic traffic filtering. You can configure up to 100 IPv6 prefix lists.

An IPv6 prefix list is composed of one or more conditional statements that pose an action (permit or deny) if a packet matches a specified prefix. In prefix lists with multiple statements, you can specify a sequence number for each statement.The specified sequence number determines the order in which the statement appears in the prefix.

You can configure an IPv6 prefix list on a global basis, then use it as input to other commands or processes, such as route aggregation, route redistribution, route distribution, route maps, and so on. When a router interface sends or receives an IPv6 packet, it applies the statement(s) within the IPv6 prefix list in their order of appearance to the packet. As soon as a match occurs, the router takes the specified action (permit or deny the packet) and stops further comparison for that packet.

You can use permit statements in the prefix list to specify the traffic that you want to send to the other feature. If you use deny statements, the traffic specified by the deny statements is not supplied to the other feature.

To configure an IPv6 prefix list and use it as input to the RIPng **distribute-list** command, enter commands such as the following:

```
BigIron(config)# ipv6 prefix-list routesfor2001 permit 2001::/16
BigIron(config)# ipv6 router rip
BigIron(config-ripng-router)# distribute-list prefix-list routesfor2001 out
ethernet 3/1
```

These commands permit the inclusion of routes with the IPv6 prefix 2001::/16 in RIPng routing updates sent from Ethernet interface 3/1.

***Syntax:*** [no] ipv6 prefix-list <name> [seq <sequence-number>] deny <ipv6-prefix>/<prefix-length> | permit <ipv6-prefix>/<prefix-length> | description <string> [ge <ge-value>] [le <le-value>]

The <name> parameter specifies the prefix list name. You use this name when using the prefix list as input to command or route map.

The **seq** <seq-number> parameter is optional and specifies the IPv6 prefix list's sequence number. If you do not specify a sequence number, the software numbers them in increments of 5, beginning with prefix list entry 5. The router interprets the prefix list entries in numerical order, beginning with the lowest sequence number.

The **description** <string> parameter is a text string describing the prefix list.

The **deny** <ipv6-prefix>/<prefix-length> | **permit** <ipv6-prefix>/<prefix-length> parameters specify the action the router takes if a packet contains a route specified in this prefix list.

You must specify the <ipv6-prefix> parameter in hexadecimal using 16-bit values between colons as documented in RFC 2373.

You must specify the <prefix-length> parameter as a decimal value. A slash mark (/) must follow the <ipv6-prefix> parameter and precede the <prefix-length> parameter.

The prefix list matches only on the specified prefix/prefix length unless you use the **ge** <prefix-length> or **le** <prefix-length> parameters. (See below.)

You can specify a range of prefix lengths for prefixes that are more specific than <ipv6-prefix>/<prefix-length>.

- If you specify only **ge** <ge-value>, then the range is from <ge-value> to 128.
- If you specify only **le** <le-value>, then the range is from <le-value> to the <prefix-length> parameter.

The <ge-value> or <le-value> you specify must meet the following condition:

prefix-length < ge-value <= le-value <= 128

If you do not specify **ge** <ge-value> or **le** <le-value>, the prefix list matches only on the exact prefix you specify with the <ipv6-prefix>/<prefix-length> parameter.

To delete the prefix list entry, use the **no** form of this command.

## Displaying Prefix List Information

To display the IPv6 prefix lists configured on a router, enter the following command at any level of the CLI:

```
BigIron(config)# show ipv6 prefix-lists

ipv6 prefix-list routesfor2001: 1 entries
     seq 5 permit 2001::/16
```

***Syntax:*** show ipv6 prefix-lists [<name>]

The <name> parameter restricts the display to the specified prefix list. Specify the name of the prefix list that you want to display.

# Chapter 12
# Configuring IPV6 Multicast Features

This chapter presents the multicast features available for IPv6 routers. Multicast concepts are explained in the *Foundry Enterprise Configuration and Management Guide*.

**NOTE:** The features in this chapter are not supported on devices running Enterprise software releases.

**NOTE:** By design, IPv6 multicast is forwarded in software, which could result in high CPU usage. That CPU usage will increase as multicast traffic increases. If wire speed IPv6 Multicast is required, BigIron RX, NetIron XMR or NetIron MLX should be considered.

## Multicast Listener Discovery and Source Specific Multicast Protocols

The Multicast Listener Discovery Version 2 (MLDv2) protocol is available on Terathon devices running Terathon software release 02.0.00 and later. IPv6 routers use the MLDv2 protocol to discover multicast listeners, or nodes that wish to receive multicast packets on directly attached links. MLDv2 supports source filtering, the ability of a node to send reports on traffic that is from a specific address source or from all multicast addresses except the specified address sources. The information is then provided to the source specific multicast (SSM) routing protocols such as PIM-SSM.

The IPv6 router stores a list of multicast addresses for each attached link. For each multicast address, the IPv6 router stores a filter mode and a source list. The filter mode is set to INCLUDE if all nodes in the source list for a multicast address are in the INCLUDE state. If the filter mode is INCLUDE, then only traffic from the addresses in the source list is allowed. The filter mode is set to EXCLUDE if at least one of the nodes in the source list is in an EXCLUDE state. If the filter mode is EXCLUDE, traffic from nodes in the source list is denied and traffic from other sources is allowed.

The source list and filter mode are created when the IPv6 querier router sends a query. The querier router is the one with the lowest source IPv6 address. It sends out any of the following queries:

- General query – The querier sends this query to learn all multicast addresses that need to be listened to on an interface.
- Address specific query – The querier sends this query to determine if a specific multicast address has any listeners.
- Address specific and source specific query – The querier sends this query to determine if specified sources of a specific multicast address have any listeners.

In response to these queries, multicast listeners send the following reports:

- Current state – This report specifies the source list for a multicast address and whether the filter mode for that

source list is INCLUDE or EXCLUDE.

- Filter-mode change – This report specifies if there has been a change to the filter mode for the source list and provides a new source list.
- Source list change – This report specifies the changes to the source list.

MLDv1 is compatible with IGMPv2 and MLDv2 is compatible with IGMPv3.

## Enabling MLDv2

MLDv1 is enabled once PIM Sparse (PIM-SM) is enabled on an interface. You then enable version 2 of MLD, the version that supports source filtering.

MLDv2 interoperates with MLDv1. MLDv1 messages are understood by MLDv2. When an IPv6 router detects that the node is operating in MLDv1 mode, the router switches to MLDv1 for that node even though queries are sent in MLDv2.

To enable PIM-SM, enter the following command at the interface level:

```
NetIron 40G(config-if-e10000-1/1)#ipv6 pim-sparse
```

***Syntax:*** [no] ipv6 pim-sparse

Once PIM-SM is enabled, specify which version of MLD will be used by entering the following command:

```
NetIron 40G (config-if-e10000-1/1)#ipv6 mld port-version 2
```

***Syntax:*** ipv6 mld port-version <version-number>

Enter 1 or 2 for <version-number>. Be sure to enter “2” if you want to use source filtering.

### Enabling Source Specific Multicast

Once MDLv2 is enabled, source specific multicast for PIM can be enabled for multicast group addresses in the ff30::0/16 IPv6 address range. If MLDv2 is enabled, but SSM is not, the IPv6 router builds the Shortest Path Tree (SPT) as well as the shared (RP) tree and produces a (*.G) record. If both MLDv2 and SSM are enabled, then only the SPT is built to produce a (S,G) record.

To enable SSM on a Foundry device running PIM-SM, enter commands such as the following:

```
BigIronBigIron(config)# ipv6 router pim
BigIronBigIron(config-ipv6-pim-router)# ssm-enable
```

***Syntax:*** [no] ssm-enable

Enter the **ssm-enable** command under the IPv6 router PIM level to globally enable source specific multicast filtering.

### Setting the Query Interval

You can define the frequency at which MLD query messages are sent. For example, if you want queries to be sent every 50 seconds, enter a command such as the following:

```
BigIronBigIron(config)#ipv6 mld query-interval 50
```

***Syntax:*** ipv6 mld query-interval <seconds>

Specify 1 – 3600 for <seconds>. The default is 60 seconds.

### Setting the Maximum Response Time

You can define the maximum amount of time a multicast listener has to respond to queries by entering a command such as the following:

```
BigIronBigIron(config)#ipv6 mld max-response-time 5
```

***Syntax:*** ipv6 mld max-response-time <seconds>

Specify 1 – 64 for <seconds>. The default is 5 seconds.

### Setting the Last Listener Query Count

The Last Listener Query Count is the number of Multicast-Address- Specific Queries sent before the router assumes there are no remaining listeners for an address on a link. You can set the last listener query count by entering a command such as the following:

```
BigIron(config)#ipv6 mld llqc 5
```

***Syntax:*** ipv6 mld llqc <seconds>

Specify 2 – 7 for <seconds>.

### Setting the Last Listener Query Interval

The Last Listener Query Interval is the Maximum Response Delay inserted into Multicast-Address-Specific Queries sent in response to Done messages, and is also the amount of time between Multicast- Address-Specific Query messages. When the device receives an MLDv1 leave message or an MLDv2 state change report, it sends out a query and expects a response within the time specified by this value. Using a lower value allows members to leave groups more quickly. You can set the last listenener query interval by entering a command such as the following:

```
BigIron(config)#ipv6 mld llqi 5
```

***Syntax:*** ipv6 mld llqi <seconds>

Specify 1 – 10 for <seconds>.

### Setting the Robustness

You can specify the number of times that the router sends each MLD message from this interface. Use a higher value to ensure high reliability from MLD. You can set the robustness by entering a command such as the following:

```
BigIron(config)#ipv6 mld robustness 3
```

***Syntax:*** ipv6 mld robustness <seconds>

Specify 2 – 7 for <seconds>. Default is 2

### Setting the Version

You can use this command to set the MLD version (1 or 2) globally. You can select the version of MLD by entering a command such as the following:

```
BigIron(config)#ipv6 mld version 2
```

***Syntax:*** #ipv6 mld version <version-number>

Enter 1or 2 for <version-number> Default version 2

### Specifying a Port Version

At the interface level, you can specify the MLD version for a physical port within a virtual interface. You can set the version by entering a command such as the following at the interface level:

```
BigIron(config-vif-401)#ipv6 mld port-ver 1 eth 3/1
```

***Syntax:*** mld port-ver <version-number>

Specify 1 or 2 for <version-number>.

### Specifying a Static Group

A multicast group is usually learned when an MLDv1 report is received. You can configure static group membership without having to receive an MLDv1 report. To configure a virtual interface , enter a command such as the following at the interface level:

```
BigIron(config-vif-401))#ipv6 mld static-group ffe0::4c8 eth 3/1
```

To configure a Static Group on a physical interface, enter a command such as the following at the interface level:

```
BigIron(config-if-e1000-5/23)#ipv6 mld static-group ff01::6f
```

***Syntax:*** ipv6 mld static-group <multicast-group-address> [ethernet <port-number> [ethernet <port-number> | to <port-number>]* ]

Enter the IPv6 multicast group address for the <multicast-group-address>.

Enter number of the port that will be included in this static group for the ethernet <port-number> parameter. The asterisk (*) in the syntax above means that you can enter as many port numbers as you want to include in the static group. For a virtual routing interface (ve), specify the physical Ethernet ports on which to add the group address.

## Setting the Interface MLD Version

You can use this command to set the MLD version (1 or 2) for the interface. You can select the version of MLD by entering a command such as the following at the interface level:

```
BigIron(config-lbif-1)#ipv6 mld version 2
```

***Syntax:*** #ipv6 mld version <version-number>

Enter 1or 2 for <version-number>. Default is version 2

## Displaying MLD Information

The sections below present the show commands for MLD.

### *Displaying MLD group information*

To display the list of multicast groups, enter a command such as the following:

```
NetIron BigIron#show ipv6 mld group
Interface e6/18 has 11 groups
     group                                phy-port static querier life mode

1    ff33::6:b:1                          e6/18    no     yes     0    incl
2    ff33::6:a:1                          e6/18    no     yes     0    incl
3    ff33::6:9:1                          e6/18    no     yes     0    incl
4    ff33::6:8:1                          e6/18    no     yes     0    incl
5    ff33::6:7:1                          e6/18    no     yes     0    incl
6    ff33::6:6:1                          e6/18    no     yes     0    incl
7    ff33::6:5:1                          e6/18    no     yes     0    incl
8    ff33::6:4:1                          e6/18    no     yes     0    incl
9    ff33::6:3:1                          e6/18    no     yes     0    incl
10   ff33::6:2:1                          e6/18    no     yes     0    incl
11   ff33::6:1:1                          e6/18    no     yes     0    incl
```

| This Field... | Displays... |
|---|---|
| Interface <port-number> has x groups | This message shows the ID of the interface and how many multicast groups it has. |
| # | Index for the MLD group. |
| ipv6 address | IPv6 address of the multicast group. |
| phy-port | The physical port to which the group belongs. |
| static | Indicates if the group is a static group or not. |
| querier | Indicates if the multicast group is a querier or not |

| This Field... | Displays... |
|---|---|
| life | The number of seconds the interface can remain in its current mode. |
| mode | Indicates if the filter mode of the multicast group is in INCLUDE or EXCLUDE |

***Syntax:*** show ipv6 mld group

### *Displaying MLD definitions for an interface*

To display the MLD parameters on an interface, including the various timers, the current querying router, and whether or not MLD is enabled, enter the following command:

```
BigIron BigIron#show ipv6 mld interface
version = 2, query int = 125, max resp time = 10, group mem time = 635
robustness = 5, other querier present time = 630
last listener query int = 1, last listener query count = 5
e5/18: default V2, PIM sparse, addr=fe80::20c:dbff:fe80:5251
has 50 groups, Querier, default V2
group: ff1e::619, exclude, permit 0, exclude-time=513, deny 0
group: ff1e::618, exclude, permit 0, exclude-time=513, deny 0
group: ff1e::617, exclude, permit 0, exclude-time=513, deny 0
group: ff1e::616, exclude, permit 0, exclude-time=513, deny 0
group: ff1e::5a8, exclude, permit 0, exclude-time=513, deny 0
group: ff1e::5a7, exclude, permit 0, exclude-time=513, deny 0
group: ff1e::5a6, exclude, permit 0, exclude-time=513, deny 0
```

| This Field... | Displays... |
|---|---|
| version | Version of the MLD being used. |
| query int | Query interval in seconds. |
| max resp time | Number of seconds multicast groups have to respond to queries. |
| group mem time | Number of seconds multicast groups can be members of this group before aging out. |
| (details) | The following is displayed for each interface:<br>• The port ID<br>• The default MLD version being used<br>• The multicast protocol used<br>• IPV6 address of the multicast interface<br>• If the interface has groups, the group source list, IPv6 multicast address, and the filter mode are displayed. |

***Syntax:*** show ipv6 mld interface <slot> | <portnum> | loopback <num> | ve <num>

Enter the port's number if you want to display MLD information for a specific interface.

The **ethernet** <slot> | <portnum> | **loopback** <num> | **ve** <num> parameter specifies the interface.

- Enter **ethernet** <slot> | <portnum> for a physical interface (port).
- Enter **ve** <num> for a virtual interface.
- Enter **loopback** <num> for a loopback interface.

### *Displaying MLD Traffic*

To display information on MLD traffic, enter a command such as the following:

```
BigIronBigIron#show ipv6 traffic
Recv  QryV1 QryV2 G-Qry GSQry MbrV1 MbrV2 Leave IS_IN IS_EX 2_IN 2_EX ALLO  BLK
e3/1      0     0     0     0     0     0     0     0     0    0    0    0    0
e3/2      0     0     0     0     0     0     0     0     0    0    0    0    0
e6/18     0     0     0     0     0   176     0   110     0    0    0   66    0
e6/19     0     0     0     0     0   176     0   110     0    0    0   66    0
e6/20     0     0     0     0     0   176     0   110     0    0    0   66    0
e6/25     0     0     0     0     0   176     0   110     0    0    0   66    0
l1        0     0     0     0     0     0     0     0     0    0    0    0    0
Send  QryV1 QryV2 G-Qry GSQry
e3/1      0     0     0     0
e3/2      0     0     0     0
e6/18     0    10    10     0
e6/19     0    10    10     0
e6/20     0    10    10     0
e6/25     0    10    10     0
l1        0     0     0     0
R2#
```

The report has a Receive and a Send section. These sections show the following information:

| This Field | Displays |
|---|---|
| QryV1 | Number of general MLDv1 queries received or sent by the virtual routing interface. |
| QryV2 | Number of general MLDv2 queries received or sent by the virtual routing interface. |
| G-Qry | Number of group specific queries received or sent by the virtual routing interface. |
| GSQry | Number of source specific queries received or sent by the virtual routing interface. |
| MbrV1 | Number of MLDv1 membership reports received. |
| MbrV2 | Number of MLDv2 membership reports received. |
| Leave | Number of MLDv1 "leave" messages on the interface. (See 2_Ex for MLDv2.) |
| Is_IN | Number of source addresses that were included in the traffic. |
| Is_EX | Number of source addresses that were excluded in the traffic. |
| 2_IN | Number of times the interface mode changed from exclude to include. |

| This Field | Displays |
|---|---|
| 2_EX | Number of times the interface mode changed from include to exclude. |
| ALLOW | Number of times that additional source addresses were allowed or denied on the interface. |
| BLK | Number of times that sources were removed from an interface. |

***Syntax:*** show ipv6 mld traffic

### Clearing IPv6 MLD Traffic

To clear statistics on IPv6 MLD traffic, enter the following command:

```
BigIronBigIron(config)# clear ipv6 mld traffic ethernet 7/10
```

***Syntax:*** clear ipv6 mld traffic ethernet <slot-number> | <port-number> | ve <ve-number>

Select **ethernet** and enter the interface's slot number and port number to clear MLD traffic on a physical interface.

Select **ve** and enter the virtual routing interface's number to clear MLD traffic from a virtual routing interface.

# Chapter 13
# Managing a Foundry Device Over IPv6

You can perform the following management tasks on Foundry devices that support IPv6:



**NOTE:** Foundry FES devices can serve as managment hosts on an IPv6 network. However, IPv6 routing functionality is not supported for these devices. For information about the IPv6 features supported for these devices, see "Supported IPv6 Management Features" on page 13-12

This chapter describes the IPv6 management command syntax, and provides examples. It does not describe the existing command syntax for IPv4. For information about the IPv4-related command syntax, see the *Foundry Switch and Router Command Line Interface Reference*.

# IPv6 Access List

When you enter the **ipv6 access-list** command, the Foundry device enters the IPv6 Access List configuration level, where you can access several commands for configuring IPv6 ACL entries. For information about these commands, see Chapter 10, “Configuring an IPv6 Access Control List” .

**NOTE:** Unlike IPv4, there is no distinction between standard and extended ACLs in IPv6.

**EXAMPLES:**

```
BigIron(config)# ipv6 access-list netw
BigIron(config-ipv6-access-list-netw)#
```

***Syntax:*** [no] ipv6 access-list <acl name>

The <acl name> parameter specifies a name for the IPv6 ACL. An IPv6 ACL name cannot start with a numeral, for example, 1access. Also, an IPv4 ACL and an IPv6 ACL cannot share the same name.

# IPv6 Copy

The **copy** command for IPv6 allows you to do the following:

- Copy a file from a specified source to an IPv6 TFTP server
- Copy a file from an IPv6 TFTP server to a specified destination

## Copying a File to an IPv6 TFTP Server

You can copy a file from the following sources to an IPv6 TFTP server:

- Flash memory
- Running configuration
- Startup configuration

### Copying a File from Flash Memory

For example, to copy the primary or secondary boot image from the device’s flash memory to an IPv6 TFTP server, enter a command such as the following:

```
BigIron# copy flash tftp 2001:7382:e0ff:7837::3 test.img secondary
```

This command copies the secondary boot image named test.img from flash memory to a TFTP server with the IPv6 address of 2001:7382:e0ff:7837::3.

***Syntax:*** copy flash tftp <ipv6-address> <source-file-name> primary | secondary

The <ipv6-address> parameter specifies the address of the TFTP server. You must specify this address in hexadecimal using 16-bit values between colons as documented in RFC 2373.

The <source-file-name> parameter specifies the name of the file you want to copy to the IPv6 TFTP server.

The **primary** keyword specifies the primary boot image, while the **secondary** keyword specifies the secondary boot image.

### Copying a File from the Running or Startup Configuration

For example, to copy the running configuration to an IPv6 TFTP server, enter a command such as the following:

```
BigIron# copy running-config tftp 2001:7382:e0ff:7837::3 newrun.cfg
```

This command copies the running configuration to a TFTP server with the IPv6 address of 2001:7382:e0ff:7837::3 and names the file on the TFTP server newrun.cfg.

***Syntax:*** copy running-config | startup-config tftp <ipv6-address> <destination-file-name>

Specify the **running-config** keyword to copy the running configuration file to the specified IPv6 TFTP server.

Specify the **startup-config** keyword to copy the startup configuration file to the specified IPv6 TFTP server.

The tftp <ipv6-address> parameter specifies the address of the TFTP server. You must specify this address in hexadecimal using 16-bit values between colons as documented in RFC 2373.

The <destination-file-name> parameter specifies the name of the file that is copied to the IPv6 TFTP server.

## Copying a File from an IPv6 TFTP Server

You can copy a file from an IPv6 TFTP server to the following destinations:

- Flash memory
- Running configuration
- Startup configuration

### Copying a File to Flash Memory

For example, to copy a boot image from an IPv6 TFTP server to the primary or secondary storage location in the device's flash memory, enter a command such as the following:

```
BigIron# copy tftp flash 2001:7382:e0ff:7837::3 test.img secondary
```

This command copies a boot image named test.img from an IPv6 TFTP server with the IPv6 address of 2001:7382:e0ff:7837::3 to the secondary storage location in the device's flash memory.

***Syntax:*** copy tftp flash <ipv6-address> <source-file-name> primary | secondary

The <ipv6-address> parameter specifies the address of the TFTP server. You must specify this address in hexadecimal using 16-bit values between colons as documented in RFC 2373.

The <source-file-name> parameter specifies the name of the file you want to copy from the IPv6 TFTP server.

The **primary** keyword specifies the primary storage location in the device's flash memory, while the **secondary** keyword specifies the secondary storage location in the device's flash memory.

### Copying a File to the Running or Startup Configuration

For example, to copy a configuration file from an IPv6 TFTP server to the router's running or startup configuration, enter a command such as the following.

```
BigIron# copy tftp running-config 2001:7382:e0ff:7837::3 newrun.cfg overwrite
```

This command copies the newrun.cfg file from the IPv6 TFTP server and overwrites the router's running configuration file with the contents of newrun.cfg.

---

**NOTE:** To activate this configuration, you must reload (reset) the device.

---

***Syntax:*** copy tftp running-config | startup-config <ipv6-address> <source-file-name> [overwrite]

Specify the **running-config** keyword to copy the running configuration from the specified IPv6 TFTP server.

Specify the **startup-config** keyword to copy the startup configuration from the specified IPv6 TFTP server.

The <ipv6-address> parameter specifies the address of the TFTP server. You must specify this address in hexadecimal using 16-bit values between colons as documented in RFC 2373.

The <source-file-name> parameter specifies the name of the file that is copied from the IPv6 TFTP server.

The **overwrite** keyword specifies that the device should overwrite the current configuration file with the copied file. If you do not specify this parameter, the device copies the file into the current running or startup configuration but does not overwrite the current configuration.

# IPv6 Ncopy

The **ncopy** command for IPv6 allows you to do the following:

- Copy a primary or secondary boot image from flash memory to an IPv6 TFTP server.
- Copy the running configuration to an IPv6 TFTP server.
- Copy the startup configuration to an IPv6 TFTP server
- Upload various files from an IPv6 TFTP server.

## Copying a Primary or Secondary Boot Image from Flash Memory to an IPv6 TFTP Server

For example, to copy the primary or secondary boot image from the device's flash memory to an IPv6 TFTP server, enter a command such as the following:

```
BigIron# ncopy flash primary tftp 2001:7382:e0ff:7837::3 primary.img
```

This command copies the primary boot image named primary.img from flash memory to a TFTP server with the IPv6 address of 2001:7382:e0ff:7837::3.

***Syntax:*** ncopy flash primary | secondary tftp <ipv6-address> <source-file-name>

The **primary** keyword specifies the primary boot image, while the **secondary** keyword specifies the secondary boot image.

The **tftp** <ipv6-address> parameter specifies the address of the TFTP server. You must specify this address in hexadecimal using 16-bit values between colons as documented in RFC 2373.

The <source-file-name> parameter specifies the name of the file you want to copy from flash memory.

## Copying the Running or Startup Configuration to an IPv6 TFTP Server

For example, to copy a device's running or startup configuration to an IPv6 TFTP server, enter a command such as the following:

```
BigIron# ncopy running-config tftp 2001:7382:e0ff:7837::3 bakrun.cfg
```

This command copies a device's running configuration to a TFTP server with the IPv6 address of 2001:7382:e0ff:7837::3 and names the destination file bakrun.cfg.

***Syntax:*** ncopy running-config | startup-config tftp <ipv6-address> <destination-file-name>

Specify the **running-config** keyword to copy the device's running configuration or the **startup-config** keyword to copy the device's startup configuration.

The **tftp** <ipv6-address> parameter specifies the address of the TFTP server. You must specify this address in hexadecimal using 16-bit values between colons as documented in RFC 2373.

The <destination-file-name> parameter specifies the name of the running configuration that is copied to the IPv6 TFTP server.

## Uploading Files from an IPv6 TFTP Server

You can upload the following files from an IPv6 TFTP server:

- Primary boot image.
- Secondary boot image.
- Running configuration.
- Startup configuration.

### Uploading a Primary or Secondary Boot Image from an IPv6 TFTP Server

For example, to upload a primary or secondary boot image from an IPv6 TFTP server to a device's flash memory, enter a command such as the following:

```
BigIron# ncopy tftp 2001:7382:e0ff:7837::3 primary.img flash primary
```

This command uploads the primary boot image named primary.img from a TFTP server with the IPv6 address of 2001:7382:e0ff:7837::3 to the device's primary storage location in flash memory.

***Syntax:*** ncopy tftp <ipv6-address> <source-file-name> flash primary | secondary

The **tftp** <ipv6-address> parameter specifies the address of the TFTP server. You must specify this address in hexadecimal using 16-bit values between colons as documented in RFC 2373.

The <source-file-name> parameter specifies the name of the file you want to copy from the TFTP server.

The **primary** keyword specifies the primary location in flash memory, while the **secondary** keyword specifies the secondary location in flash memory.

### Uploading a Running or Startup Configuration from an IPv6 TFTP Server

For example to upload a running or startup configuration from an IPv6 TFTP server to a device, enter a command such as the following:

```
BigIron# ncopy tftp 2001:7382:e0ff:7837::3 newrun.cfg running-config
```

This command uploads a file named newrun.cfg from a TFTP server with the IPv6 address of 2001:7382:e0ff:7837::3 to the device.

***Syntax:*** ncopy tftp <ipv6-address> <source-file-name> running-config | startup-config

The **tftp** <ipv6-address> parameter specifies the address of the TFTP server. You must specify this address in hexadecimal using 16-bit values between colons as documented in RFC 2373.

The <source-file-name> parameter specifies the name of the file you want to copy from the TFTP server.

Specify the **running-config** keyword to upload the specified file from the IPv6 TFTP server to the device. The device copies the specified file into the current running configuration but does not overwrite the current configuration.

Specify the **startup-config** keyword to upload the specified file from the IPv6 TFTP server to the device. The the device copies the specified file into the current startup configuration but does not overwrite the current configuration.

## IPv6 Debug

The **debug ipv6** commands enable the collection of information about IPv6 configurations for troubleshooting.

***Syntax:*** debug ipv6 <address> <cache> <icmp> <mld> <nd> <packet> <ra>

- address - IPv6 address
- cache - IPv6 cache entry
- icmp - ICMPv6
- mld - MLD protocol activity
  - <add-del-oif>[<all><clear>] <clear> <detail> <down-port> <error> <group> <level> <mcache-group> <mcache-source> <packet> <phy-port> <prime-port> <show> <source> <timer> <vlan>
- nd - neighbor discovery
- packet - IPv6 packet
- ra - router add

# IPv6 HTTP and HTTPS

When you have an IPv6 management station connected to a switch with an IPv6 address applied to the management port, you can manage the switch with the GUI browser by entering **http://<ipv6 address> or https://<ipv6 address>** in the browser address field.

**NOTE:** IPv6 HTTP and HTTPS are not supported in BigIron software release 08.0.00.

# IPv6 Logging

This feature allows you to specify an IPv6 server as the Syslog server.

## Specifying an IPv6 Syslog Server

To specify an IPv6 Syslog server, enter the log host ipv6 command as shown below:

**EXAMPLES:**

```
FES Switch(config)# log host ipv6 2000:2383:e0bb::4/128
```

***Syntax:*** [no] log host ipv6 <ipv6-address> [<udp-port-num>]

The <ipv6-address> you specify must be in hexadecimal using 16-bit values between colons as documented in RFC 2373.

The <udp-port-num> optional parameter specifies the UDP application port used for the Syslog facility.

**Possible values:** See above.

**Default value:** N/A

# Name-to-IPv6 Address Resolution using IPv6 DNS Server

The Domain Name Server (DNS) resolver feature lets you use a host name to performTelnet, ping, and traceroute commands. You can also define a DNS domain on a Foundry device and thereby recognize all hosts within that domain. After you define a domain name, the Foundry device automatically appends the appropriate domain to the host and forwards it to the domain name server.

For example, if the domain “newyork.com” is defined on a Foundry device, and you want to initiate a ping to host “NYC01” on that domain, you need to reference only the host name in the command instead of the host name and its domain name. For example, you could enter either of the following commands to initiate the ping:

```
BigIron# ping nyc01
BigIron# ping nyc01.newyork.com
```

## Defining a DNS Entry

You can define up to four DNS servers for each DNS entry. The first entry serves as the primary default address. If a query to the primary address fails to be resolved after three attempts, the next gateway address is queried (also up to three times). This process continues for each defined gateway address until the query is resolved. The order in which the default gateway addresses are polled is the same as the order in which you enter them.

Suppose you want to define the domain name of newyork.com on a Foundry device and then define four possible default DNS gateway addresses. To do so using IPv4 addressing, you would enter the following commands:

```
BigIron(config)# ip dns domain-name newyork.com
BigIron(config)# ip dns server-address 209.157.22.199 205.96.7.15 208.95.7.25
201.98.7.15
```

***Syntax:*** ip dns server-address <ip-addr> [<ip-addr>] [<ip-addr>] [<ip-addr>]

In this example, the first IP address in the **ip dns server-address...** command becomes the primary gateway address and all others are secondary addresses. Because IP address 201.98.7.15 is the last address listed, it is also the last address consulted to resolve a query.

### Defining an IPv6 DNS Entry

IPv6 defines new DNS record types to resolve queries for domain names to IPv6 addresses, as well as IPv6 addresses to domain names. Foundry devices running IPv6 software support AAAA DNS records, which are defined in RFC 1886.

AAAA DNS records are analogous to the A DNS records used with IPv4. They store a complete IPv6 address in each record. AAAA records have a type value of 28.

To establish an IPv6 DNS entry for the device, enter the following command:

```
BigIron(config)# ipv6 dns domain-name companynet.com
```

***Syntax:*** [no] ipv6 dns domain-name <domain name>

To define an IPv6 DNS server address, enter the following command:

```
BigIron(config)# ipv6 dns server-address 200::1
```

***Syntax:*** [no] ipv6 dns server-address <ipv6-addr> [<ipv6-addr>] [<ipv6-addr>] [<ipv6-addr>]

As an example, in a configuration where ftp6.companynet.com is a server with an IPv6 protocol stack, when a user pings ftp6.companynet.com, the Foundry device attempts to resolve the AAAA DNS record. In addition, if the DNS server does not have an IPv6 address, as long as it is able to resolve AAAA records, it can still respond to DNS queries.

## IPv6 Ping

The **ping** command allows you to verify the connectivity from a Foundry device to an IPv6 device by performing an ICMP for IPv6 echo test.

For example, to ping a device with the IPv6 address of 2001:3424:847f:a385:34dd::45 from the Foundry device, enter the following command:

```
BigIron# ping ipv6 2001:3424:847f:a385:34dd::45
```

***Syntax:*** ping ipv6 <ipv6-address> [outgoing-interface [<port> | ve <number>]] [source <ipv6-address>] [count <number>] [timeout <milliseconds>] [ttl <number>] [size <bytes>] [quiet] [numeric] [no-fragment] [verify] [data <1-to-4 byte hex>] [brief]

The <ipv6-address> parameter specifies the address of the router. You must specify this address in hexadecimal using 16-bit values between colons as documented in RFC 2373.

The **outgoing-interface** keyword specifies a physical interface over which you can verify connectivity. If you specify a physical interface, such as an Ethernet interface, you must also specify the port number of the interface. If you specify a virtual interface, such as a VE, you must specify the number associated with the VE.

The **source** <ipv6-address> parameter specifies an IPv6 address to be used as the origin of the ping packets.

The **count** <number> parameter specifies how many ping packets the router sends. You can specify from 1 - 4294967296. The default is 1.

The **timeout** <milliseconds> parameter specifies how many milliseconds the router waits for a reply from the pinged device. You can specify a timeout from 1 - 4294967296 milliseconds. The default is 5000 (5 seconds).

The **ttl** <number> parameter specifies the maximum number of hops. You can specify a TTL from 1 - 255. The default is 64.

The **size** <bytes> parameter specifies the size of the ICMP data portion of the packet. This is the payload and does not include the header. You can specify from 0 - 4000. The default is 16.

The **no-fragment** keyword turns on the "don't fragment" bit in the IPv6 header of the ping packet. This option is disabled by default.

The **quiet** keyword hides informational messages such as a summary of the ping parameters sent to the device and instead only displays messages indicating the success or failure of the ping. This option is disabled by default.

The **verify** keyword verifies that the data in the echo packet (the reply packet) is the same as the data in the echo request (the ping). By default the device does not verify the data.

The **data** <1 - 4 byte hex> parameter lets you specify a specific data pattern for the payload instead of the default data pattern, "abcd", in the packet's data payload. The pattern repeats itself throughout the ICMP message (payload) portion of the packet.

**NOTE:** For parameters that require a numeric value, the CLI does not check that the value you enter is within the allowed range. Instead, if you do exceed the range for a numeric value, the software rounds the value to the nearest valid value.

The **brief** keyword causes ping test characters to be displayed. The following ping test characters are supported:

**!** Indicates that a reply was received.

**.** Indicates that the network server timed out while waiting for a reply.

**U** Indicates that a destination unreachable error PDU was received.

**I** Indicates that the user interrupted ping.

# Restricting Web Access

You can restrict Web management access to management funcitons only on a Foundry device that is acting as an IPv6 host, or restrict access to the Foundry host to a single IPv6 device.

## Restricting Web Management Access by Specifying an IPv6 ACL

You can specify an IPv6 ACL that restricts Web management access to management functions on the Foundry device that is acting as the IPv6 host. For example:

```
FES Switch(config)# access-list 12 deny host 2000:2383:e0bb::2/128 log
FES Switch(config)# access-list 12 deny 30ff:3782::ff89/128 log
FES Switch(config)# access-list 12 deny 3000:4828::fe19/128 log
FES Switch(config)# access-list 12 permit any
FES Switch(config)# web access-group ipv6 12
```

***Syntax:*** web access-group ipv6 <ipv6 ACL name>

where <ipv6 ACL name> is a valid IPv6 ACL.

## Restricting Web Management Access to an IPv6 Host

You can specify a single device with an IPv6 address to have Web management access to the Foundry host device. No other device except the one with the specified IPv6 address can access the Foundry device's Web management interface. For example:

```
FES Switch(config)# web client ipv6 3000:2383:e0bb::2/128
```

***Syntax:*** web client ipv6 <ipv6-address>

The <ipv6-address> you specify must be in hexadecimal format using 16-bit values between colons as documented in RFC 2373.

**NOTE:** This feature is not supported for BigIron software release 08.0.00.

# SNMP over IPv6

## Restricting SNMP Access to an IPv6 Node

You can restrict SNMP access to the Foundry device (including IronView Network Manager) to the IPv6 host whose IP address you specify. To do so, enter a command such as the following:

```
FES Switch(config)# snmp-client ipv6 2001:efff:89::23
```

***Syntax:*** snmp-client ipv6 <ipv6-address>

The <ipv6-address> you specify must be in hexadecimal format using 16-bit values between colons as documented in RFC 2373.

**NOTE:** This feature is not supported for BigIron software release 08.0.00.

## Specifying an IPv6 Host as an SNMP Trap Receiver

You can specify an IPv6 host as a trap receiver to ensure that all SNMP traps sent by the Foundry device will go to the same SNMP trap receiver or set of receivers, typically one or more host devices on the network. To do so, enter a command such as the following:

```
FES Switch(config)# snmp-server host ipv6 2001:efff:89::13
```

***Syntax:*** snmp-server host ipv6 <ipv6-address>

The <ipv6-address> you specify must be in hexadecimal format using 16-bit values between colons as documented in RFC 2373.

**NOTE:** This feature is not supported for BigIron software release 08.0.00.

**Possible values:** N/A

**Default value:** N/A

# Secure Shell

Secure Shell (SSH) is a mechanism that allows secure remote access to management functions on the Foundry device. SSH provides a function similar to Telnet. You can log into and configure the Foundry device using a publicly or commercially available SSH client program, just as you can with Telnet. However, unlike Telnet, which provides no security, SSH provides a secure, encrypted connection to the Foundry device.

For information about configuring SSH on the Foundry device, see the *Foundry Security Guide*.

To open an SSH session from an IPv6 host running an SSH client program to the Foundry device, open your SSH client program and specify the IPv6 address of the router.

# IPv6 Telnet

Telnet sessions can be established between a Foundry device to a remote IPv6 host, and from a remote IPv6 host to the Foundry device using IPv6 addresses.

## Using the IPv6 Telnet Command

The **telnet** command establishes a Telnet connection from a Foundry device to a remote IPv6 host using the console. Up to five ***read-access*** Telnet sessions are supported on the router at one time. ***Write-access*** through Telnet is limited to one session, and only one outgoing Telnet session is supported on the router at one time. To see the number of open Telnet sessions at any time, enter the **show telnet** command.

**EXAMPLES:**

To establish a Telnet connection to a remote host with the IPv6 address of 3001:2837:3de2:c37::6, enter the following command:

```
BigIron# telnet 3001:2837:3de2:c37::6
```

***Syntax:*** telnet <ipv6-address> [<port-number> | outgoing-interface ethernet <port> | ve <number>]

The <ipv6-address> parameter specifies the address of a remote host. You must specify this address in hexadecimal using 16-bit values between colons as documented in RFC 2373.

The <port-number> parameter specifies the port number on which the Foundry device establishes the Telnet connection. You can specify a value between 1 - 65535. If you do not specify a port number, the Foundry device establishes the Telnet connection on port 23.

If the IPv6 address you specify is a link-local address, you must specify the **outgoing-interface** ethernet <port> | ve <number> parameter. This parameter identifies the interface that must be used to reach the remote host. If you specify an Ethernet interface, you must also specify the port number associated with the interface. If you specify a VE interface, also specify the VE number.

### Establishing a Telnet Session From an IPv6 Host

To establish a Telnet session from an IPv6 host to the Foundry device, open your Telnet application and specify the IPv6 address of the Layer 3 Switch.

## IPv6 Traceroute

The **traceroute** command allows you to trace a path from the Foundry device to an IPv6 host.

The CLI displays trace route information for each hop as soon as the information is received. Traceroute requests display all responses to a minimum TTL of 1 second and a maximum TTL of 30 seconds. In addition, if there are multiple equal-cost routes to the destination, the Foundry device displays up to three responses.

For example, to trace the path from the Foundry device to a host with an IPv6 address of 3301:23dd:349e:a384::34, enter the following command:

```
BigIron# traceroute ipv6 3301:23dd:349e:a384::34
```

***Syntax:*** traceroute ipv6 <ipv6-address>

The <ipv6-address> parameter specifies the address of a host. You must specify this address in hexadecimal using 16-bit values between colons as documented in RFC 2373.

## Viewing IPv6 SNMP Server Addresses

Starting with the releases shown above, many of the existing **show** commands now display IPv6 addresses for IPv6 SNMP servers. The following shows an example output for the **show snmp server** command.

```
SW-FES2402 Switch#show snmp server

      Contact:
     Location:
Community(ro): .....

Traps
              Warm/Cold start: Enable
                     Link up: Enable
                   Link down: Enable
               Authentication: Enable
     Locked address violation: Enable
         Power supply failure: Enable
                  Fan failure: Enable
          Temperature warning: Enable
                 STP new root: Enable
          STP topology change: Enable
                         vsrp: Enable

  Total Trap-Receiver Entries: 4

Trap-Receiver IP-Address                Port-Number Community
     1        192.147.201.100                162     .....
     2        4000::200                      162     .....
     3        192.147.202.100                162     .....
     4        3000::200                      162     .....
```

## Disabling Router Advertisement and Solicitation Messages

Router advertisement and solicitation messages enable a node on a link to discover the routers on the same link. By default, router advertisement and solicitation messages are permitted on the Foundry device. To disable these messages, configure an IPv6 access list that denies them. The following shows an example configuration.

**EXAMPLES:**

```
FES Switch(config)# ipv6 access-list rtradvert
FES Switch(config)# deny icmp any any router-advertisement
FES Switch(config)# deny icmp any any router-solicitation
FES Switch(config)# permit ipv6 any any
```

### Disabling IPv6 on a Layer 2 Switch

IPv6 is enabled by default in the Foundry Layer 2 switch code. If desired, you can disable IPv6 on a global basis on an Foundry device running the switch code. To do so, enter the following command at the Global CONFIG level of the CLI:

```
FES Switch(config)# no ipv6 enable
```

***Syntax:*** no ipv6 enable

To re-enable IPv6 after it has been disabled, enter **ipv6 enable**.

**NOTE:** IPv6 is disabled by default and must be configured on each interface that will support it.

# IPv6 Management Support for FES Devices

You can configure Foundry FES devices as management hosts. The devices will have IPv6 addresses on the interfaces, but will not have IPv6 routing enabled. For FES devices, IPv6 management is available on devices running Layer 2, Base Layer 3, or Full Layer 3 software images.

## Supported IPv6 Management Features

The following IPv6 management features are supported starting in the releases shown above:

- Automatic address configuration is supported in the Foundry Layer 2 switch code and the Foundry Layer 3 router code. (Automatic configuration of an IPv6 global address is supported only if there is an IPv6 router present on the network.)
- Manual IPv6 address configuration
- HTTP/HTTPS over IPv6
- IPv6 ping
- Telnet using IPv6
- TFTP using IPv6
- IPv6 Traceroute
- Name-to-IPv6 address resolution using IPv6 DNS Server
- IPv6 access list
- IPv6 debug
- SSH version 1 over IPv6
- SNMP over IPv6
- Logging (Syslog) over IPv6

## Unsupported IPv6 Features

The following IPv6 features are not supported on FES devices:

- IPv6 routing
- IPv6Tunneling
- IPv6 security
- IPv6 in boot PROM
- IPv6 address configuration using DHCP
- IPv6 TFTP using IPv6 link local address for a TFTP server
- IPv6 link local address is not supported for IPv6 DNS server
- TACAS, RADIUS, NTP over IPv6

## IPv6 Feature Differences between Layer 2 and Layer 3 Devices

A few of the features in Foundry Layer 2 switch code differ from their counterparts in Foundry Layer 3 router code for FastIron devices. The differences are as follows:

- The router code supports automatic configuration of IPv6 link local and IPv6 addresses per interface. The switch code supports automatic configuration of IPv6 link local and IPv6 address on a global basis only.

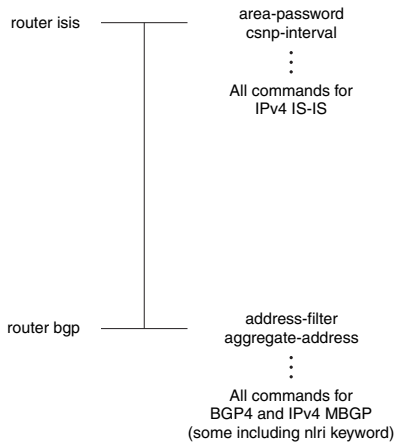
**NOTE:** Automatic configuration of an IPv6 global address is supported only if there is an IPv6 router on the network.

- IPv6 is enabled by default in the Layer 2 switch code. If desired, you can disable IPv6 on a global basis on Foundry devices running the switch code. IPv6 is disabled by default in the router code and must be configured on each interface that will support IPv6.
- Some **configuration**, **show**, **clear**, and **debug** CLI commands are available in the router code only.

# Appendix A
# Global and Address Family Configuration Levels

This chapter describes the restructured CLI for BGP and IS-IS on Foundry devices that support IPv6. In earlier versions of Foundry software, the CLI for BGP4 and IPv4 IS-IS is structured as shown in Figure A.1.

**Figure A.1Earlier Structure of BGP4 and IPv4 IS-IS CLI**




To configure BGP4 and IPv4 MBGP, you enter the **router bgp** command, which allows you to enter the BGP router configuration level. At this level, you can access commands that allow you to configure all aspects of BGP4 and IPv4 MBGP, including those that configure the protocol itself and those that configure unicast routes and multicast routes. (To configure aspects of multicast routes, you specify the **nlri** keyword with a command.)

To configure IPv4 IS-IS, you enter the **router isis** command, which allows you to enter the IS-IS router configuration level. At this level, you can access commands that allow you to configure all aspects of IPv4 IS-IS, including those that configure the protocol itself and those that configure unicast routes.

In both cases, it is up to the router to sort out, for example, if commands entered at the BGP router configuration level apply to BGP4 itself, to BGP4 unicast routes, or to IPv4 MBGP routes.

The introduction of IPv6 in this software version necessitates the restructuring of the existing BGP4 and IPv4 IS-IS CLI for the following reasons:

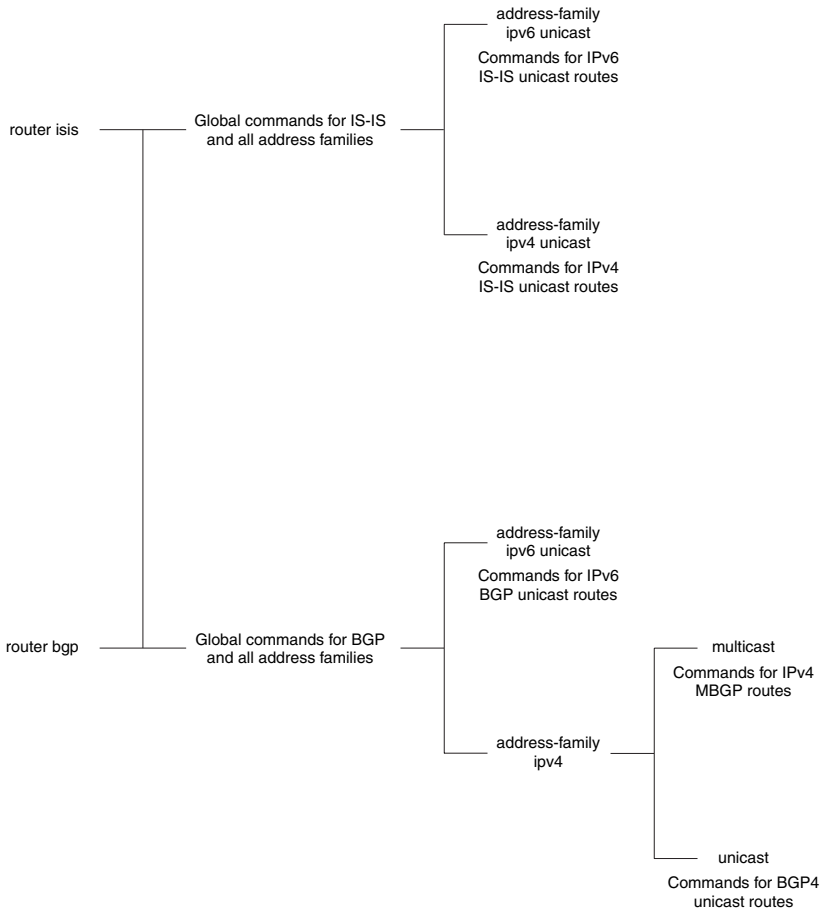
- To accommodate the IPv6-related CLI.
- To simplify the configuration of BGP4 unicast and IPv4 MBGP routes.

The CLI in this software version includes two new layers of CLI for BGP and IS-IS (see Figure A.2). The new layers are as follows:

- A global layer for the configuration of the BGP and IS-IS protocols themselves.
- Address families that separate the configuration of the following:
  - IPv6 and IPv4
  - Routing protocol

  Sub-address families separate the configuration of the following:
  - Unicast routes
  - Multicast routes

**Figure A.2New Structure of IPv4 and BGP4+ and IS-IS CLI**




# Accessing the Address Family Configuration Level

For example, to access the BGP4 multicast address family configuration level, enter the following command while at the global BGP configuration level:

```
BigIron(config-bgp)# address-family ipv4 multicast
BigIron(config-bgp-ipv4m)#
```

***Syntax:*** address-family ipv4 unicast | ipv4 multicast | ipv6 unicast

The `(config-bgp-ipv4m)#` prompt indicates that you are at the IPv4 multicast address family configuration level. While at this level, you can access commands that allow you to configure BGP4 multicast routes. The

commands that you enter at this level apply to BGP4 multicast routes only. You can generate a configuration for BGP4 multicast routes that is separate and distinct from configurations for BGP4 unicast routes and BGP4+ unicast routes.

**NOTE:** Each address family configuration level allows you to access commands that apply to that particular address family only. To enable a feature in a particular address family, you must specify any associated commands for that feature in that particular address family. You cannot expect the feature, which you may have configured in the BGP4 multicast address family, to work in the BGP4 unicast address family unless it is explicitly configured in the BGP4 unicast address family.

To exit from the BGP4 multicast address family configuration level, enter the following command:

```
BigIron(config-bgp-ipv4m)# exit-address-family
BigIron(config-bgp)#
```

***Syntax:*** exit-address-family

Typically, entering the **exit-address-family** command at one of the address family configuration levels returns you to the global IS-IS configuration level or the BGP4 unicast address family configuration level, which is the default BGP4 level. For backward compatibility, you can currently access commands related to BGP4 unicast routes at both global BGP4 configuration and BGP4 unicast address family configuration level. Both of these level are indicated by the `(config-bgp)#` prompt.

## Backward Compatibility for Existing BGP4 and IPv4 IS-IS Configurations

When a router is upgraded with this software version, the software automatically converts the existing BGP4 unicast and all IPv4 IS-IS configurations into the new address families. The software automatically converts some of the IPv4 MBGP configuration into the new address family. Specifically, the software does the following:

- Leaves the global BGP4 and IPv4 IS-IS configurations as is.
- Converts the configuration for BGP4 unicast neighbors and routes into the BGP4 unicast address family.
- Converts the configuration for IPv4 IS-IS unicast routes into the IPv4 IS-IS unicast address family.
- Converts the configuration for IPv4 MBGP neighbors into IPv4 MBGP address family.

**NOTE:** The software does not convert all aspects of the IPv4 MBGP configuration. You will have to reconfigure the following features: network routes, aggregate routes, redistribution of routes into IPv4 MBGP, and route map filters. You can use the **show run** and **show ip bgp config** commands to check your IPv4 MBGP configuration.

Previously, IPv4 MBGP routes were configured using commands that include the **nlri** keyword This version of software does not support the **nlri** keyword with IPv4 and IPv6 MBGP commands. From this point forward, you must use the address families to configure all versions of BGP, IS-IS, and MBGP.

## Global BGP4 Commands and BGP4 Unicast Route Commands

A global BGP command is a command that configures the BGP routing protocol and therefore applies to all IPv4 and IPv6 address families. You can access the global commands while at the global BGP configuration level.

A BGP4 unicast route command is a command that configures a BGP4 unicast route. For backward compatibility, you can currently access BGP4 unicast route commands at both global BGP4 configuration and BGP4 unicast address family configuration levels. To help you distinguish the global BGP4 commands from the BGP4 unicast route commands that you access at the global BGP4 configuration level, this section provides a listing of global BGP commands.

The following are global BGP commands:

- **address-filter**
- **always-compare-med**

- **as-path-filter**
- **as-path-ignore**
- **bgp-redistribute-internal**
- **cluster-id**
- **community-filter**
- **compare-routerid**
- **confederation identifier**
- **confederation peers**
- **default-local-preference**
- **distance**
- **enforce-first-as**
- **fast-external-fallover**
- **ignore-invalid-confed-as-path**
- **local-as**
- **med-missing-as-worst**
- **timers**

The following are global BGP commands for configuring peer groups and neighbors:

- **neighbor** <ipv4-address> | <ipv6-address> | <peer-group-name> **advertisement-interval**
- **neighbor** <ipv4-address> | <ipv6-address> | <peer-group-name> **description**
- **neighbor** <ipv4-address> | <ipv6-address> | <peer-group-name> **distribute-list acl_name in**
- **neighbor** <ipv4-address> | <ipv6-address> | <peer-group-name> **distribute-list acl_name out**
- **neighbor** <ipv4-address> | <ipv6-address> | <peer-group-name> **distribute-list in**
- **neighbor** <ipv4-address> | <ipv6-address> | <peer-group-name> **distribute-list out**
- **neighbor** <ipv4-address> | <ipv6-address> | <peer-group-name> **ebgp-multihop**
- **neighbor** <ipv4-address> | <ipv6-address> | <peer-group-name> **filter-list in**
- **neighbor** <ipv4-address> | <ipv6-address> | <peer-group-name> **filter-list out**
- **neighbor** <ipv4-address> | <ipv6-address> | <peer-group-name> **next-hop-self**
- **neighbor** <ipv4-address> | <ipv6-address> | <peer-group-name> **password**
- **neighbor** <ipv4-address> | <ipv6-address> | <peer-group-name> **peer-group**
- **neighbor** <ipv4-address> | <ipv6-address> | <peer-group-name> **remote-as**
- **neighbor** <ipv4-address> | <ipv6-address> | <peer-group-name> **remove-private-as**
- **neighbor** <ipv4-address> | <ipv6-address> | <peer-group-name> **shut_down**
- **neighbor** <ipv4-address> | <ipv6-address> | <peer-group-name> **soft-reconfiguration inbound**
- **neighbor** <ipv4-address> | <ipv6-address> | <peer-group-name> **timers**
- **neighbor** <ipv4-address> | <ipv6-address> | <peer-group-name> **update-source**

An address family command is a command that modifies the behavior of BGP for a specific address family. The following are address family commands:

- **aggregate-address**

- **auto-summary** (applies to the IPv4 unicast address family only)
- **client-to-client-reflection**
- **dampening**
- **default-information-originate**
- **default-metric**
- **maximum-paths**
- **multipath**
- **neighbor** <ipv4-address> | <ipv6-address> | <peer-group-name> **filter-list in** (applies to the IPv4 unicast address family only)
- **network**
- **next-hop-enable-default**
- **next-hop-recursion** (applies to the IPv4 unicast address family only)
- **readvertise** (applies to the IPv4 unicast address family only)
- **redistribute**
- **table-map**
- **update-time**

The following commands configure policies for neighbors or peer groups for a specific address family:

- **neighbor** <ipv4-address> | <ipv6-address> | <peer-group-name> **activate**
- **neighbor** <ipv4-address> | <ipv6-address> | <peer-group-name> **capability orf prefixlist**
- **neighbor** <ipv4-address> | <ipv6-address> | <peer-group-name> **default-originate**
- **neighbor** <ipv4-address> | <ipv6-address> | <peer-group-name> **filter-list as-path-access-list in**
- **neighbor** <ipv4-address> | <ipv6-address> | <peer-group-name> **filter-list as-path-access-lis out**
- **neighbor** <ipv4-address> | <ipv6-address> | <peer-group-name> **maximum-prefix**
- **neighbor** <ipv4-address> | <ipv6-address> | <peer-group-name> **prefix-list prefix_list_name in**
- **neighbor** <ipv4-address> | <ipv6-address> | <peer-group-name> **prefix-list prefix_list_name out**
- **neighbor** <ipv4-address> | <ipv6-address> | <peer-group-name> **route-map in**
- **neighbor** <ipv4-address> | <ipv6-address> | <peer-group-name> **route-map out**
- **neighbor** <ipv4-address> | <ipv6-address> | <peer-group-name> **route-reflector-client**
- **neighbor** <ipv4-address> | <ipv6-address> | <peer-group-name> **send-community**
- **neighbor** <ipv4-address> | <ipv6-address> | <peer-group-name> **unsuppress-map**
- **neighbor** <ipv4-address> | <ipv6-address> | <peer-group-name> **weight**

**NOTE:** Currently, you can create a neighbor with an IPv4 or IPv6 address at the global BGP configuration/IPv4 unicast address family configuration level. For example, if you create a neighbor with an IPv4 address at this level, by default, the neighbor is enabled to exchange IPv4 unicast prefixes. However, this neighbor cannot exchange IPv4 multicast prefixes until you explicitly enable it to do so by entering the **neighbor** <ipv4-address> | <peer-group-name> **activate** command at the IPv4 multicast address family configuration level. Likewise, if you create a neighbor with an IPv6 address at this level, the neighbor will not exchange IPv6 unicast prefixes until you explicitly enable it to do so by entering the **neighbor** <ipv6-address> | <peer-group-name> **activate** command at the IPv6 unicast address family configuration level.

Also, if you create a neighbor at the IPv4 multicast address family configuration or IPv6 unicast address family

configuration levels, by default, the neighbor is enabled to exchange IPv4 multicast prefixes or IPv6 unicast prefixes, respectively. You do not need to explicitly enable the neighbor by entering the **neighbor** <ipv4-address> | <ipv6-address> | <peer-group-name> **activate** command at either level.

# Appendix B
# Supported IPv6 RFCs and Internet Drafts

Table B.1 lists the features supported by Foundry's implementation of IPv6 and the RFC associated with each feature.

**Table B.1: Supported IPv6 Features and Associated RFCs and Internet Drafts**

| IPv6 Feature | RFC or Internet Draft |
|---|---|
| **Services:** | |
| IPv6 | 2460 |
| IPv6 addressing architecture | 3513 |
| IPv6 unicast address allocation architecture | 1887 |
| IPv6 aggregatable global unicast address format | 2374 |
| IPv6 multicast address assignments | 2375 |
| Proposed TLA and NLA Assignment Rules | 2450 |
| IPv6 testing address allocation | 2471 |
| Reserved IPv6 subnet anycast address | 2526 |
| Initial IPv6 sub-TLA ID assignments | 2928 |
| IPv6 router alert option | 2711 |
| ICMPv6 | 2463 |
| Neighbor discovery/ICMPv6 redirect | 2461 |
| IPv6 stateless autoconfiguration/IPv6 duplicate address detection | 2462 |
| Path maximum transmission unit (MTU) discovery for IPv6 | 1981 |
| Static cache entry for IPv6 neighbor discovery | None |
| IPv6 standard access control list | None |

**Table B.1: Supported IPv6 Features and Associated RFCs and Internet Drafts**

| IPv6 Feature | RFC or Internet Draft |
|---|---|
| IPv6 ICMP rate limiting | None |
| **Transition mechanisms for IPv6 Hosts and Routers:** | |
| Connection of IPv6 domains via IPv4 clouds (automatic 6to4 tunnels for IPv6) | 3056 |
| IPv6 manually configured tunnels/Automatic IPv4-compatible tunnels | 2893 |
| **Data link layer protocols:** | |
| Ethernet, Fast Ethernet, Gigabit Ethernet, and 10-Gigabit Ethernet | 2464 |
| **Routing protocols:** | |
| Static routes | None |
| RIPng | 2080 |
| OSPF version 3 | 2740 |
| IPv6 IS-IS | draft-ietf-isis-ipv6 |
| IPv6 MBGP extensions/link-local address peering | 2545 |
| Route redistribution | None |
| **Management SNMP MIBs:** | |
| IP Version 6 Management Information Base for the Transmission Control Protocol | 2452 |
| IP Version 6 Management Information Base for the User Datagram Protocol | 2454 |
| Management Information Base for IP Version 6: Textual Conventions and General Group | 2465 |
| Management Information Base for IP Version 6: ICMPv6 Group | 2466 |