CONFIGURATION GUIDE

Extreme Network OS Layer 3 Routing Configuration Guide, 7.3.0

Supporting Network OS 7.3.0

9035242-01
15 June 2018

Contents

Preface

Document conventions

The document conventions describe text formatting conventions, command syntax conventions, and important notice formats used in Extreme technical documentation.

Notes, cautions, and warnings

Notes, cautions, and warning statements may be used in this document. They are listed in the order of increasing severity of potential hazards.

NOTE
A Note provides a tip, guidance, or advice, emphasizes important information, or provides a reference to related information.

ATTENTION
An Attention statement indicates a stronger note, for example, to alert you when traffic might be interrupted or the device might reboot.

CAUTION
A Caution statement alerts you to situations that can be potentially hazardous to you or cause damage to hardware, firmware, software, or data.

DANGER
A Danger statement indicates conditions or situations that can be potentially lethal or extremely hazardous to you. Safety labels are also attached directly to products to warn of these conditions or situations.

Text formatting conventions

Text formatting conventions such as boldface, italic, or Courier font may be used to highlight specific words or phrases.

Format	Description
bold text	Identifies command names.
	Identifies keywords and operands.
	Identifies the names of GUI elements.
	Identifies text to enter in the GUI.
italic text	Identifies emphasis.
	Identifies variables.
	Identifies document titles.
`Courier font`	Identifies CLI output.

Format	Description
	Identifies command syntax examples.

Command syntax conventions

Bold and italic text identify command syntax components. Delimiters and operators define groupings of parameters and their logical relationships.

Convention	Description
bold text	Identifies command names, keywords, and command options.
italic text	Identifies a variable.
[ ]	Syntax components displayed within square brackets are optional. Default responses to system prompts are enclosed in square brackets.
{ **x** | **y** | **z** }	A choice of required parameters is enclosed in curly brackets separated by vertical bars. You must select one of the options.
x | **y**	A vertical bar separates mutually exclusive elements.
< >	Nonprinting characters, for example, passwords, are enclosed in angle brackets.
...	Repeat the previous element, for example, *member*[*member*...].
\	Indicates a "soft" line break in command examples. If a backslash separates two lines of a command input, enter the entire command at the prompt without the backslash.

Extreme resources

Visit the Extreme website to locate related documentation for your product and additional Extreme resources.

White papers, data sheets, and the most recent versions of Extreme software and hardware manuals are available at www.extremenetworks.com. Product documentation for all supported releases is available to registered users at www.extremenetworks.com/support/documentation.

Document feedback

Quality is our first concern at Extreme, and we have made every effort to ensure the accuracy and completeness of this document. However, if you find an error or an omission, or you think that a topic needs further development, we want to hear from you.

You can provide feedback in two ways:

- Use our short online feedback form at http://www.extremenetworks.com/documentation-feedback-pdf/
- Email us at internalinfodev@extremenetworks.com

Provide the publication title, part number, and as much detail as possible, including the topic heading and page number if applicable, as well as your suggestions for improvement.

Contacting Extreme Technical Support

As an Extreme customer, you can contact Extreme Technical Support using one of the following methods: 24x7 online or by telephone. OEM customers should contact their OEM/solution provider.

If you require assistance, contact Extreme Networks using one of the following methods:

- GTAC (Global Technical Assistance Center) for immediate support
  - Phone: 1-800-998-2408 (toll-free in U.S. and Canada) or +1 408-579-2826. For the support phone number in your country, visit: www.extremenetworks.com/support/contact.
  - Email: support@extremenetworks.com. To expedite your message, enter the product name or model number in the subject line.
- GTAC Knowledge - Get on-demand and tested resolutions from the GTAC Knowledgebase, or create a help case if you need more guidance.
- The Hub - A forum for Extreme customers to connect with one another, get questions answered, share ideas and feedback, and get problems solved. This community is monitored by Extreme Networks employees, but is not intended to replace specific guidance from GTAC.
- Support Portal - Manage cases, downloads, service contracts, product licensing, and training and certifications.

Before contacting Extreme Networks for technical support, have the following information ready:

- Your Extreme Networks service contract number and/or serial numbers for all involved Extreme Networks products
- A description of the failure
- A description of any action(s) already taken to resolve the problem
- A description of your network environment (such as layout, cable type, other relevant environmental information)
- Network load at the time of trouble (if known)
- The device history (for example, if you have returned the device before, or if this is a recurring problem)
- Any related RMA (Return Material Authorization) numbers

About this document

Supported hardware and software

In those instances in which procedures or parts of procedures documented here apply to some devices but not to others, this guide identifies exactly which devices are supported and which are not.

Although many different software and hardware configurations are tested and supported by Extreme Networks, Inc. for Network OS, documenting all possible configurations and scenarios is beyond the scope of this document.

The following hardware platforms are supported by this release of Network OS:

- ExtremeSwitching VDX 2741
- ExtremeSwitching VDX 2746
- ExtremeSwitching VDX 6740
  - ExtremeSwitching VDX 6740-48
  - ExtremeSwitching VDX 6740-64
- ExtremeSwitching VDX 6740T
  - ExtremeSwitching VDX 6740T-48
  - ExtremeSwitching VDX 6740T-64
  - ExtremeSwitching VDX 6740T-1G
- ExtremeSwitching VDX 6940-36Q
- ExtremeSwitching VDX 6940-144S
- ExtremeSwitching VDX 8770
  - ExtremeSwitching VDX 8770-4
  - ExtremeSwitching VDX 8770-8

To obtain information about a Network OS version other than this release, refer to the documentation specific to that version.

Using the Network OS CLI

For complete instructions and support for using the Extreme Network OS command line interface (CLI), refer to the *Extreme Network OS Command Reference*.

What's new in this document

NOTE

On October 30, 2017, Extreme Networks, Inc. acquired the data center networking business from Brocade Communications Systems, Inc. This document has been updated to remove or replace references to Brocade Communications, Inc. with Extreme Networks., Inc., as appropriate.

This document supports the following features introduced in Network OS 7.3.0:

- Appendix-E implementation for OSPFv2 Type7 LSA
- IP best route management

For complete information about this release, refer to the *Network OS Release Notes*.

ARP and ND Scaling Enhancements

ARP and Neighbor Discovery

When forwarding traffic, a device needs to know the destination's MAC address, because each IP packet is encapsulated in an ethernet frame. The MAC address is needed not only for the packet's final destination but also for a next hop towards the destination.

The technology by which a device gets the MAC address varies between IPv4 and IPv6, as follows:

- IPv4: Address Resolution Protocol (ARP)
- IPv6: Neighbor Discovery (ND)

IPv4 traffic

When the destination's IP address is known, to get the MAC address, a device first searches its ARP cache. A match for the IP address supplies the corresponding MAC address. Otherwise, the device broadcasts an ARP request. The network devices receive such ARP requests, and the host with a matching IP address sends an ARP reply that includes its MAC address.

IPv6 traffic

When the destination's IPv6 address is known, to get the MAC address, a device first searches its neighbor cache. A match for the IPv6 address supplies the corresponding MAC address. Otherwise, the device broadcasts a neighbor solicitation (NS) request. The network devices receive the NS request, and the host with a matching IPv6 address sends a neighbor advertisement (NA) reply that includes its MAC address.

ARP and ND scaling enhancements

In many network configurations, ARP and Neighbor Discovery (ND) traffic and caching consume significant resources, which can be optimized.

ARP and ND suppression

In a data center fabric, the ARP and ND suppression features can help reduce ARP and ND control traffic.

> NOTE
> This feature is supported on VLANs.

The default scenario (disablement of ARP and ND suppression) can lead to excess control traffic:

1. A device needs to forward traffic to an IP address, but the corresponding MAC address is not in its ARP or ND cache.
2. The device broadcasts an ARP or ND request throughout the IP Fabric.

When ARP and ND suppression are enabled, excess control traffic is reduced:

1. A device needs to forward traffic to an IP address, but the corresponding MAC address is not in its ARP or ND cache.
2. The device broadcasts an ARP or ND request.
3. The leaf BGP EVPN control plane intercepts the request and looks for a match in its local cache.
   - If there is a match, the control plane responds to the device (rather than broadcasting the original request to the entire IP Fabric).
   - Only if there is no match, does the leaf control plane broadcast the request to the entire IP Fabric.

   NOTE
   When the static anycast gateway feature is enabled in an IP Fabric, to prevent ARP/ND broadcast across the network, the user should enable ARP/ND suppression on the respective VLANs.

Supported platforms

The ARP and ND suppression features are supported on the following platforms:

- ExtremeSwitching VDX 6740 series
- ExtremeSwitching VDX 6940 series
- ExtremeSwitching VDX 2741 (not supported for IP Fabrics)
- ExtremeSwitching VDX 2746 (not supported for IP Fabrics)

Enabling ARP and ND suppression on a VLAN

Use this procedure to enable and disable ARP and ND suppression on a VLAN.

1. Enter **configure terminal** to access global configuration mode.

   ```
   device# configure terminal
   ```

2. Enter the **interface vlan** command to access VLAN configuration mode.

   ```
   device(config)# interface vlan 110
   ```

3. To enable ARP suppression, enter **suppress-arp**.

   ```
   device(config-Vlan-110)# suppress-arp
   ```

   To disable ARP suppression, enter **no suppress-arp**.

4. To enable ND suppression, enter **suppress-nd**.

   ```
   device(config-Vlan-110)# suppress-nd
   ```

   To disable ND suppression, enter **no suppress-nd**.

The following example enables ARP suppression on VLAN 110.

```
device# configure terminal
device(config)# interface vlan 110
device(config-Vlan-110)# suppress-arp
```

Enabling and disabling ARP learning

Use this procedure to enable ARP learning not only locally, but from all ARP requests. ARP learning decreases the time needed to populate the ARP cache.

1. Enter **configure terminal** to access global configuration mode.

```
device# configure terminal
```

2. Enter the **rbridge-id** command to access RBridge ID configuration mode.

```
device(config)# rbridge-id 1
```

3. Enter the **interface ve** command to access VE configuration mode.

```
device(config-rbridge-id-1)# interface ve 110
```

4. To enable fabric learning, enter the **ip arp learn-any** command.

```
device(config-ve-110)# ip arp learn-any
```

   To disable fabric learning, enter the **no ip arp learn-any** command.

ARP and ND suppression show and clear commands

There is a full range of ARP and ND suppression **show** and **clear** commands, listed here with descriptions.

TABLE 1 ARP and ND suppression show commands in the *Command Reference*

Command	Description
show ip arp suppression-cache	Displays IPv4 ARP suppression information.
show ip arp suppression-statistics	Displays IPv4 ARP suppression statistics.
show ip arp suppression-status	Displays the IPv4 ARP suppression status.
show ipv6 nd suppression-cache	Displays IPv6 ND suppression information.
show ipv6 nd suppression-statistics	Displays IPv6 ND suppression statistics.
show ipv6 nd suppression-status	Displays the IPv6 ND suppression status.

TABLE 2 ARP and ND suppression clear commands in the *Command Reference*

Command	Description
clear ip arp suppression-cache	Clears the IPv4 ARP suppression cache. You can also clear the cache for a specified VLAN.
clear ip arp suppression-statistics	Clears the IPv4 ARP suppression statistical information. You can also clear statistics for a specified VLAN.
clear ipv6 nd suppression-cache	Clears the IPv6 ND suppression cache. You can also clear the cache for a specified VLAN.
clear ipv6 nd suppression-statistics	Clears the IPv6 ND suppression statistical information. You can also clear statistics for a specified VLAN.

Conversational ARP and ND

Conversational ARP and ND reduce the number of cached ARP and ND entries by programming only active flows into the forwarding plane. This feature helps to optimize utilization of hardware resources.

In many use-case scenarios—especially in an IP Fabric—there are software requirements for ARP and ND entries beyond the hardware capacity. Conversational ARP and ND limit storage-in-hardware to active ARP and ND entries; aged-out entries are deleted automatically.

By default, the aging-out threshold is 300 seconds. (You can change the threshold to any integer value from 60 through 100,000 seconds, either before or during enablement.) Any entry that does not have at least one conversation before aging-out is deleted from the cache. Each conversation restarts the clock for that entry.

However, aging-out is also influenced by the enablement and disablement cycle:

1. When you enable conversational ARP and ND, a fast-aging policy of 30 seconds (not configurable) applies to all entries in the ARP and ND caches at that time.
2. From enablement of conversational ARP and ND, the aging-time value applies for existing entries with new traffic and for new entries.
3. Upon disablement, the conversational ARP and ND timers no longer apply. All current entries become permanent as do all new entries.

The following entries are not subject to conversational behavior:

- Static ARPs and NDs
- Next hops

Supported platforms

Conversational ARP and ND are supported on the following platforms:

- ExtremeSwitching VDX 6740 series
- ExtremeSwitching VDX 6940 series
- ExtremeSwitching VDX 2741 (not supported for IP Fabrics)
- ExtremeSwitching VDX 2746 (not supported for IP Fabrics)

Enabling and disabling conversational ARP and ND

Conversational ARP and ND can reduce the number of cached ARP and ND entries, optimizing utilization of hardware resources.

1. Enter **configure terminal** to access global configuration mode.

   ```
   device# configure terminal
   ```

2. Enter the **rbridge-id** command to access RBridge ID configuration mode.

   ```
   device(config)# rbridge-id 1
   ```

3. To specify an aging-time value other than the default 300 seconds, enter the **host-table aging-time conversational** command.

   ```
   device(config-rbridge-id-1)# host-table aging-time conversational 600
   ```

   To restore the default aging-time value of 300 seconds, enter the **no host-table aging-time conversational** command.

4. To enable conversational ARP and ND, enter the **host-table aging-mode conversational** command.

```
device(config-rbridge-id-1)# host-table aging-mode conversational
```

To disable conversational ARP and ND, enter the **no host-table aging-mode conversational** command.

The following example implements conversational ARP and ND. The current aging-time value applies.

```
device# configure terminal
device(config)# rbridge-id 1
device(config-rbridge-id-1)# host-table aging-mode conversational
```

Conversational ARP and ND show and clear commands

Conversational ARP and ND **show** and **clear** commands are listed here with descriptions.

TABLE 3 Conversational ARP and ND show commands in the *Command Reference*

Command	Description
show arp slot	Displays the ARP cache.
show ipv6 neighbor slot	Displays IPv6 neighbor information.

TABLE 4 Conversational ARP and ND clear commands in the *Command Reference*

Command	Description
clear arp	Clears the local ARP cache and then resends ARP requests to the local hosts.
clear arp no-refresh	Clears the ARP cache, without resending ARP requests to the local hosts.
clear ipv6 neighbor	Clears the IPv6 ND cache and then resends ND requests to the local hosts.
clear ipv6 neighbor no-refresh	Clears the ND cache, without resending ND requests to the local hosts.

Dynamic ARP Inspection (DAI)

Dynamic ARP inspection (DAI) overview

Dynamic ARP inspection (DAI) is a security feature that validates address resolution protocol (ARP) packets in a subnet, and discards packets with invalid IP/MAC address bindings.

Address resolution protocol (ARP)

When forwarding traffic, a device needs to know the destination's MAC address, because each IP packet is encapsulated in a MAC packet. The MAC address is needed not only for the packet's final destination but also for a next hop towards the destination.

When the destination's IP address is known, to get the MAC address a device first searches its ARP cache. A match for the IP address supplies the corresponding MAC address. Otherwise, the device broadcasts an ARP request. The devices on the subnet receive such ARP requests, and the host whose IP address matches sends an ARP reply that includes its MAC address.

ARP poisoning

An ARP poisoning attack, also known as ARP spoofing, targets the ARP caches of devices connected to the subnet, with the goal of intercepting traffic.

A malicious host might use one of the following tactics:

- Send ARP packets claiming to have an IP address that actually belongs to another host.
- Reply to an ARP request with its own MAC address, thereby causing other hosts on the subnet to store this information in their ARP tables, even replacing an existing ARP entry.
- Send gratuitous replies without having received any ARP requests.

If the poisoning succeeds, traffic intended for the device under attack is instead routed to the attacker computer. The attacker has various options:

- Not forward any traffic to the computer under attack or forward some of the traffic, but not all of it (denial-of-service attacks).
- Forward inspected traffic to the compromised device (interception).
- Modify the traffic and then forward it (man-in-the-middle attack).

Dynamic ARP inspection (DAI)

On VLANs, dynamic ARP inspection (DAI) can examine incoming ARP packets. DAI discards packets with invalid IP/MAC address bindings, guarding against ARP-poisoning attacks; only valid ARP requests and responses are relayed.

Towards enabling DAI, you need to decide which ports you are defining as trusted and which as untrusted. ARP packets on trusted ports bypass all DAI validations and are forwarded as required. DAI examines ARP packets only on untrusted ports.

NOTE
DAI is currently supported only in non-DHCP environments. You specify valid, static IP/MAC address bindings in the **permit** statements of ARP ACLs.

When DAI is implemented on a VLAN, it monitors untrusted ports as follows:

- Intercepts ARP requests and responses
- Compares the IP/MAC address bindings with the **permit** statements in the ACL applied to the VLAN
- Drops invalid packets
- Forwards valid packets to the appropriate destination, with the option of generating log entries

DAI implementation

Address Resolution Protocol (ARP) access control lists (ACLs), applied to untrusted ports, permit only ARP packets with specified IP/MAC address bindings. Such ACLs implement Dynamic ARP Inspection (DAI).

DAI configuration guidelines

Follow these guidelines when implementing ARP ACLs for DAI.

- DAI is available on Layer 2 VLANs and Layer 3 virtual Ethernets (VE).

  NOTE
  In these guidelines, the term VLAN can also include VEs (where relevant).

- VLANs supported for DAI include:
  - 802.1Q VLANs
  - VLANs that cross multiple VCS clusters
  - Virtual fabric (VFAB) VLANs
  - VE interfaces under virtual routing and forwarding (VRF). Both default and non-default VRFs are supported.
- DAI is not supported for management interfaces or Layer 3 router interfaces.
- On a VLAN with DAI enabled, the following types of member ports are supported for DAI:
  - Physical interfaces (in switchport mode)
  - Port-channel interfaces (LAGs or vLAGs) (in switchport mode)
- In a VCS VLAN with multiple RBridges, enabling DAI on the primary RBridge automatically implements it on all of the VLAN RBridges.
- On DAI-enabled VLANs, the rate limit per chip is 64 ARP packets per second (pps).

Creating an ARP ACL

Use this procedure to create an ARP ACL and **permit ip host** rules.

1. Enter **configure terminal** to access global configuration mode.

   ```
   device# configure terminal
   ```

2. Enter the **arp access-list** command to create the ACL.

   ```
   device(config)# arp access-list arpACL1
   ```

3. For each ACL rule, enter a **permit ip host** command.

```
device(config-arp-acl)# permit ip host 1.1.1.1 mac host 0020.2222.2222
device(config-arp-acl)# permit ip host 1.1.1.2 mac host 0020.2222.2223
```

Applying an ARP ACL to a VLAN

Use this procedure to apply an ARP ACL to a VLAN.

NOTE
To replace an ARP ACL on a VLAN, there is no need to remove a previously applied ACL. The most recent ACL applied replaces any previous ACL.

1. Enter **configure terminal** to change to global configuration mode.

```
device# configure terminal
```

2. Enter the **interface vlan** command to access the VLAN.

```
device(conf)# interface vlan 200
```

3. Enter the **ip arp inspection filter** command, specifying the ACL.

```
device(conf-if-vlan-200)# ip arp inspection filter ARP_ACL_01
```

4. To return to global configuration mode—for example, to define trusted/untrusted interfaces and to enable DAI—enter **exit**.

```
device(conf-if-vlan-200)# exit
```

Defining trusted and untrusted interfaces under DAI

Use this procedure to specify untrusted and trusted interfaces under Dynamic ARP Inspection (DAI).

1. Enter **configure terminal** to change to global configuration mode.

```
device# configure terminal
```

2. For each interface that you need to define as trusted or untrusted, complete the following steps:
   a) Enter the **interface** command to access interface configuration mode.

```
device(config)# interface tengigabitethernet 1/0/1
```

   b) To define the interface as trusted, enter the **ip arp inspection trust** command.

```
device(config-if-te-1/0/1)# ip arp inspection trust
```

   c) To redefine a currently trusted interface as untrusted (default), enter the **no ip arp inspection trust** command.

```
device(config-if-te-1/0/1)# no ip arp inspection trust
```

The following example defines a port-channel interface as trusted.

```
device# configure terminal
device(config)# interface port-channel 200
device(config-Port-channel-200)# ip arp inspection trust
```

Enabling and disabling DAI

Use this procedure to enable Dynamic ARP Inspection (DAI) on a VLAN.

1. Enter **configure terminal** to change to global configuration mode.

```
device# configure terminal
```

2. Enter the **interface vlan** command to access configuration mode on the VLAN for which you are enabling DAI.

```
device(config)# interface vlan 1001
```

3. To enable DAI, enter the **ip arp inspection** command.

```
device(config-if-vlan-1001)# ip arp inspection
```

4. To disable DAI, enter the **no ip arp inspection** command.

```
device(config-if-vlan-1001)# no ip arp inspection
```

Enabling and disabling DAI logging

Use this procedure to enable Dynamic ARP Inspection (DAI) on a VLAN, with terminal logging.

1. Enter **configure terminal** to access global configuration mode.

```
device# configure terminal
```

2. Enter the **arp access-list** command to create the access list.

```
device(config)# arp access-list arpACL1
```

3. For each ACL rule, enter a **permit ip host** command.

```
device(config-arp-acl)# permit ip host 1.1.1.1 mac host 0020.2222.2222 log
device(config-arp-acl)# permit ip host 1.1.1.2 mac host 0020.2222.2223
```

NOTE

If there is no permit statement that contains the **log** keyword, DAI logging does not occur.

4. Enter the **exit** command to return to global configuration mode.

```
device(config-arp-acl)# exit
```

5. Enter the **interface vlan** command to access configuration mode on the VLAN for which you are enabling DAI.

```
device(config)# interface vlan 200
```

6. Enter the **ip arp inspection filter** command, specifying an ACL containing a **permit** statement with the **log** keyword.

```
device(config-vlan-200)# ip arp inspection filter arpACL1
```

7. Enter the **ip arp inspection logging acl-match matchlog** command.

```
device(config-vlan-200)# ip arp inspection logging acl-match matchlog
```

8. Enter the **ip arp inspection** command to enable DAI.

```
device(config-vlan-200)# ip arp inspection
```

9. Enter the **end** command to return to privileged EXEC mode.

```
device(config-vlan-200)# end
```

10. Enter the terminal monitor command.

```
device# terminal monitor
```

The following example applies ARP_ACL_01 to VLAN 200, enables DAI logging on VLAN 200, enables DAI, and displays the log.

```
device# configure terminal
device(conf)# interface vlan 200
device(config-vlan-200)# ip arp inspection filter ARP_ACL_01
device(config-vlan-200)# ip arp inspection logging acl-match
device(config-vlan-200)# ip arp inspection
device(config-vlan-200)# end
device# terminal monitor
Terminal monitoring is enabled.
device# 015/03/11-18:47:06 PERMIT: arp Packet with srcIp=1.1.1.1, srcMac=0001.0001.0001,
dstIp=10.1.1.1,dstMac=ffff.ffff.ffff on Vlan: 11
2015/03/11-18:47:06 PERMIT: arp Packet with srcIp=1.1.1.1, srcMac=0001.0001.0001,
dstIp=10.1.1.1,dstMac=ffff.ffff.ffff on Vlan: 11
2015/03/11-18:47:06 PERMIT: arp Packet with srcIp=1.1.1.1, srcMac=0001.0001.0001,
dstIp=10.1.1.1,dstMac=ffff.ffff.ffff on Vlan: 11
2015/03/11-18:47:06 PERMIT: arp Packet with srcIp=1.1.1.1, srcMac=0001.0001.0001,
dstIp=10.1.1.1,dstMac=ffff.ffff.ffff on Vlan: 11
2015/03/11-18:47:06 PERMIT: arp Packet with srcIp=1.1.1.1, srcMac=0001.0001.0001,
dstIp=10.1.1.1,dstMac=ffff.ffff.ffff on Vlan: 11
2015/03/11-18:47:06 PERMIT: arp Packet with srcIp=1.1.1.1, srcMac=0001.0001.0001,
dstIp=10.1.1.1,dstMac=ffff.ffff.ffff on Vlan: 11
2015/03/11-18:47:06 PERMIT: arp Packet with srcIp=1.1.1.1, srcMac=0001.0001.0001,
dstIp=10.1.1.1,dstMac=ffff.ffff.ffff on Vlan: 11
2015/03/11-18:47:06 PERMIT: arp Packet with srcIp=1.1.1.1, srcMac=0001.0001.0001,
dstIp=10.1.1.1,dstMac=ffff.ffff.ffff on Vlan: 11
2015/03/11-18:47:06 PERMIT: arp Packet with srcIp=1.1.1.1, srcMac=0001.0001.0001,
dstIp=10.1.1.1,dstMac=ffff.ffff.ffff on Vlan: 11
```

DAI Show/Clear commands

There is a full range of dynamic ARP inspection (DAI) show and clear commands. They are documented in the *Network OS Command Reference*, and listed here with descriptions.

TABLE 5 DAI Show commands in the Network OS Command Reference

Command	Description
show arp access-list	For one or all ARP ACLs defined on a device, displays ACL names and their **permit** statements.
show ip arp inspection interfaces	For VLANs enabled for DAI, displays a list of trusted interfaces.
show ip arp inspection statistics	Displays DAI statistics for one or more DAI-enabled VLANs.
show ip arp inspection	Displays DAI information for one or more VLANs.

TABLE 6 DAI Clear commands in the Network OS Command Reference

Command	Description
clear ip arp inspection statistics	Clears DAI statistics for all DAI-enabled VLANs.

IP Addressing

IP addressing overview

IPv4 uses a 32-bit addressing system designed for use in packet-switched networks. IPv4 routing is enabled by default on Network OS devices that operate at Layer 3 and cannot be disabled.

IPv4 is an Internet protocol used to deliver packets of data from a source to a destination across an interconnected system of networks. IPv4 uses a fixed-length 32-bit addressing system and is represented in a 4-byte dotted decimal format: x.x.x.x.

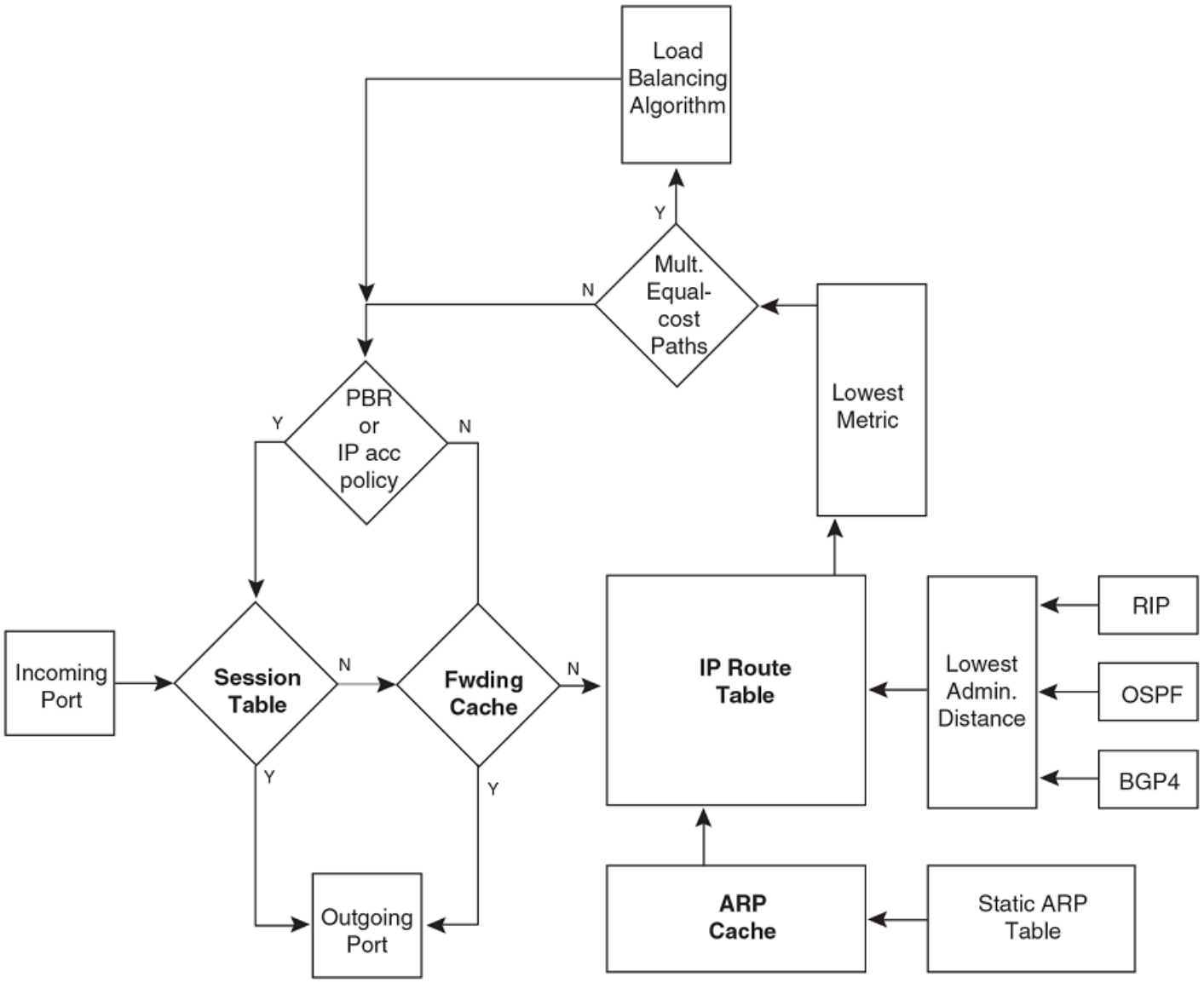
IP uses four main mechanisms to provide service:

- Type of Service (ToS)—Indicates the Quality of Service (QoS) required for a specific traffic type or network and enables a higher priority to be given to voice traffic, for example, that is more sensitive to dropped packets.
- Time to Live (TTL)—The time period for which a packet can exist before it reaches its final destination. If the TTL expires before the packet reaches its destination, the packet is destroyed. The period is set by the packet sender.
- Options—Control mechanisms such as timestamps, security, and other special routing functions that are optional.
- Header Checksum—Used to verify that the packet contents have transmitted correctly. If the checksum algorithm fails, the packet is dropped immediately.

IP does not provide a reliable communication function. No acknowledgments are sent and the only error control is the header checksum. There are not flow-control mechanisms or retransmissions. Errors detected may be reported via the Internet Control Message Protocol (ICMP).

IP packet flow through a Layer 3 device

FIGURE 1 IP Packet flow through a Layer 3 device



1. When the Layer 3 device receives an IP packet, the device checks for filters on the receiving interface. The filter may be an Access Control List (ACL) or an IP access policy. If a deny filter on the interface denies the packet, the Layer 3 device discards the packet and performs no further processing, except for generating a Syslog entry and an SNMP message, if logging is enabled for the filter.
2. If the packet is not denied at the incoming interface, the Layer 3 device looks in the session table for an entry that has the same source IP address and TCP or UDP port as the packet. If the session table contains a matching entry, the Layer 3 device immediately forwards the packet, by addressing it to the destination IP address and TCP or UDP port listed in the session table entry and sending the packet to a queue on the outgoing ports listed in the session table. The device selects the queue based on the Quality of Service (QoS) level associated with the session table entry.

3. If the session table does not contain an entry that matches the packet source address and TCP or UDP port, the Layer 3 device looks in the IP forwarding cache for an entry that matches the packet destination IP address. If the forwarding cache contains a matching entry, the device forwards the packet to the IP address in the entry. The device sends the packet to a queue on the outgoing ports listed in the forwarding cache. The device selects the queue based on the Quality of Service (QoS) level associated with the forwarding cache entry.
4. If the IP forwarding cache does not have an entry for the packet, the Layer 3 device checks the IP route table for a route to the packet destination. If the IP route table has a route, the device makes an entry in the session table or the forwarding cache, and sends the route to a queue on the outgoing ports:
   - If the running-config contains an IP access policy for the packet, the software makes an entry in the session table. The Layer 3 device uses the new session table entry to forward subsequent packets from the same source to the same destination.
   - If the running-config does not contain an IP access policy for the packet, the software creates a new entry in the forwarding cache. The Layer 3 device uses the new cache entry to forward subsequent packets to the same destination.

ARP cache and static ARP table

The ARP cache contains entries that map IP addresses to MAC addresses. A static ARP table contains entries that are user-configured.

The ARP cache entries are generally for devices that are directly attached to the Layer 3 device.

An exception is an ARP entry for an interface-based static IP route that goes to a destination that is one or more router hops away. For this type of entry, the MAC address is either the destination device MAC address or the MAC address of the router interface that answered an ARP request on behalf of the device, using proxy ARP.

ARP cache

The ARP cache can contain dynamic (learned) entries and static (user-configured) entries. The software places a dynamic entry in the ARP cache when the Layer 3 device learns a device MAC address from an ARP request or ARP reply from the device.

The software can learn an entry when the Layer 2 device or Layer 3 device receives an ARP request from another IP forwarding device or an ARP reply. Here is an example of a dynamic entry:

```
IP Address          MAC Address         Type        Age     Port
10.95.6.102        0000.00fc.ea21      Dynamic     0        6
```

Each entry contains the destination device IP address and MAC address.

Static ARP table

In addition to the ARP cache, Layer 3 devices have a static ARP table. You can add entries to the static ARP table regardless of whether or not the device the entry is for is connected to the Layer 3 device.

NOTE
Layer 3 devices have a static ARP table. Layer 2 switches do not.

The software places an entry from the static ARP table into the ARP cache when the entry interface comes up.

Here is an example of a static ARP entry.

Each entry lists the information you specified when you created the entry.

IP route table

The IP route table contains paths to IP destinations.

The IP route table can receive the paths from the following sources and it can contain leaked routes as well (from other VRFs).

- A directly-connected destination, which means there are no router hops to the destination
- A static IP route, which is a user-configured route
- A route learned through OSPF
- A route learned through BGP4

The IP route table contains the best path to a destination:

- When the software receives paths from more than one of the sources listed above, the software compares the administrative distance of each path and selects the path with the lowest administrative distance. The administrative distance is a protocol-independent value from 1 through 255.
- When the software receives two or more best paths to a destination and the paths have the same metric (cost), the software can load share traffic among the paths based on destination host or network address (based on the configuration and the Layer 3 device model).

Each IP route table entry contains the destination IP address and subnet mask and the IP address of the next-hop router interface to the destination. Each entry also indicates the port attached to the destination or the next-hop to the destination, the route IP metric (cost), and the type. The type indicates how the IP route table received the route.

IP forwarding cache

The IP forwarding cache contains entries for IP destinations, and provides a fast-path mechanism for forwarding IP packets.

When a Layer 3 device has completed the processing and addressing for a packet and is ready to forward the packet, the device checks the IP forwarding cache for an entry to the packet destination:

- If the cache contains an entry with the destination IP address, the device uses the information in the entry to forward the packet out the ports listed in the entry. The destination IP address is the address of the packet final destination. The port numbers are the ports through which the destination can be reached.
- If the cache does not contain an entry and the traffic does not qualify for an entry in the session table instead, the software can create an entry in the forwarding cache.

Each entry in the IP forwarding cache has an age timer. The age timer is not configurable.

Each IP forwarding cache entry contains the IP address of the destination, and the IP address and MAC address of the next-hop router interface to the destination. If the destination is actually an interface configured on the Layer 3 device itself, as shown here, then next-hop information indicates this. The port through which the destination is reached is also listed, as well as the VLAN and Layer 4 QoS priority associated with the destination if applicable.

NOTE

You cannot add static entries to the IP forwarding cache. You can increase the number of entries the cache can contain.

31-bit subnet masks on point-to-point networks

To conserve IPv4 address space, a 31-bit subnet mask can be assigned to point-to-point networks. Support for an IPv4 address with a 31-bit subnet mask is described in RFC 3021.

With IPv4, four IP addresses with a 30-bit subnet mask are allocated on point-to-point networks. In contrast, a 31-bit subnet mask uses only two IP addresses: all zero bits and all one bits in the host portion of the IP address. The two IP addresses are interpreted as host

addresses, and do not require broadcast support because any packet that is transmitted by one host is always received by the other host at the receiving end. Therefore, directed broadcast on a point-to-point interface is eliminated.

When the 31-bit subnet mask address is configured on a point-to-point link, using network addresses for broadcast purposes is not allowed. For example, in an IPV4 broadcast scheme, the following subnets can be configured:

- 10.10.10.1 - Subnet for directed broadcast: {*Network-number*, -1}
- 10.10.10.0 - Subnet for network address: {*Network-number*, 0}

In a point-to-point link with a 31-bit subnet mask, the previous two addresses are interpreted as host addresses and packets are not rebroadcast.

IP route management overview

IP route management is the term used to refer to software that manages routes and next hops from different sources in a routing table, from which the device selects the best routes for forwarding IP packets. This route management software gets activated automatically at system bootup and does not require preconfiguration.

IP route management runs on all platforms configured for Layer 3 and does the following:

- Maintains routes submitted by other protocols.
- Supports route redistribution.
- Supports router identification.
- Selects and synchronizes routes to the forwarding information base (FIB).
- Synchronizes the Layer 3 interface to the FIB.
- Supports the following Layer 3 interfaces: virtual ethernet (Ve), router port, loopback, and management.

  NOTE
  IP route management supports both IPv4 and IPv6 routes.

How IP route management determines best route

The sources of routes that are added into IP route management are the following:

- Dynamic routes from routing protocols. Open Shortest Path First (OSPF) and Border Gateway Protocol (BGP) are both supported.
- Static configured routes: You can add routes directly to the route table. When you add a route to the IP route table, you are creating a static IP route.
- Directly connected routes from interface configuration: When you add an IP interface, the device automatically creates a route for the network.

Administrative distance can be configured for route types other than connected routes. IP route management prefers routes with lower administrative distances.

When there is a need to originate more than one Type 7 LSA with same network address, LSA for the least specific prefix (least prefix length) uses Link State ID same as that of the network address. LSAs with more specific prefixes (higher prefix lengths) use Link State ID with their host-bits set.

Managing ECMP global configurations

The **hardware-profile** command provides options for managing Equal Cost Multiple Paths (ECMP) globally at the RBridge level.

Up to 32 ECMP paths are supported for Layer 3.

Possible options for the VDX 8770 and VDX 6940 are 8, 16, or 32; for the VDX 6740 they are 8 or 16. The default is 8 for these platforms. The following illustrates the configuration of a hardware profile.

```
device(config)# rbridge-id 1
device(config-rbridge-id-1)# hardware-profile tcam openflow
device(config-rbridge-id-1)# hardware-profile route-table ipv6-max-route maximum_paths 16 openflow on
device(config-rbridge-id-1)# hardware-profile kap default
```

IP route policy overview

IP route policy controls how routes or IP subnets are transported from one subsystem to another subsystem. The IP route policy may perform "permit" or "deny" actions so that matched routes may be allowed or denied to the target subsystem accordingly. Additionally, IP route policy may also be used for modify the characteristics of a matched route and IP subnet pair.

Two types of IP route policies are supported, *prefix-list* and *route-map*, as discussed in the following sections

IP prefix lists

An IP prefix list is identified by its name. Each IP prefix list may consist of one or more instances. The following is an example of IP prefix list "test", which is configured in RBridge ID configuration mode:

```
device# config
device(config-rbridge-id-1)#
device(config-rbridge-id-1)# ip prefix-list test deny 1.2.0.0/16 ge 17 le 30
device(config-rbridge-id-1)# ip prefix-list test permit 1.1.0.0/16
```

A matching condition of a prefix-list instance contains two portions: (1) an IP subnet prefix and(2) an optional prefix (mask) length, where **ge** (greater than or equal to) is the lower limit of the mask length, and **le** (less than or equal to) is the upper limit of the mask length. If no **ge** or **le** is given in an instance, the exact match of subnet prefix length is needed.

In the example above, a route is considered a match for instance 1 if this route is inside subnet 1.2.0.0/16 *and* whose mask length is between 17 and 30. That is, route 1.2.1.0/24 matches but route 1.2.1.1/32 does not match because of the difference in mask length.

Similar to a route map, when finding a match, each prefix-list instance is looked at in the order specified by its instance ID. The look-up terminates at the first match. A route that does not find a match in the prefix list is denied.

At present, a prefix list is not used by itself. The IP prefix list can be used as part of route-map **match** clauses. In this context, **permit** means “match” this pattern, and **deny** means “do not match this pattern."

Route maps

A route map is identified by its name. Each route map may consist of one or more instances. Each route map instance may contain zero or more **match** clauses, and zero or more **set** clauses.

At present, a route map instance represents the largest granularity of configuration. That is, the end user is required to add *and* delete route maps by means of its instance. For example, when removing a route map, an end user is required to remove this route-map in all

of its instances. A route map instance may contain more than one match condition. The overall matching condition of the instance is true only if all matching conditions are met. The following is an example of a route map:

```
device# route-map test deny 1 match interface te 0/1
device# route-map test permit 2 match ip next-hop prefix-list pre-test set tag 5000
```

In the example above, **route-map test** comprises of two instances: instance 1 denies entry for any routes whose next-hop interface is te 0/1, and instance 2 allows entry for routes whose next-hop address matches the IP subnets specified by **prefix-list pre-test** (the prefix-list instance is not shown). Additionally, each matched route has its tag set to 5000.

NOTE

The maximum number of OSPF networks that can be advertised and processed in a single area in a router is limited to 600.

A route map instance does not need to contain a matching condition; its existence implies that the matching condition for this instance is true.

A route map instance may contain more than one set clause. All **set** clauses are applied to the **match** routes when applicable.

When a route map is applied, each instance is looked at in the order specified by the instance ID. If there is a match, the instance's action are applied, and its **set** clauses are applied if the action is permitted. The search terminates at the first match. A route that does not find a match in a route map is denied.

Configuring IP route policy

Similar to ACLs, a route map and IP prefix list need to be applied for a specified policy to take effect. The following example applies a route-map to the redistribution of static routes into an OSPF domain. (For complete information on these commands, refer to the *Extreme Network OS Command Reference*.)

To set an IP route policy, perform the following steps in privileged EXEC mode.

1. Enter the **router ospf (or router bgp)** command to enable the appropriate Layer 3 protocol. This example uses OSPF and creates the route map instance "test."

   ```
   device# router ospf redistribute static route-map test area 0
   ```

2. Enter the **ip route** command to create the prefix for a static route.

   ```
   device# ip route 11.11.11.0/24 2.2.2.1
   ```

3. Enter the **ip route** command to create the next hop in the static route. Repeat as needed.

   ```
   device# ip route 11.11.11.0/24 2.2.2.2
   ```

4. Enter the **route-map** command to create the route map and prefix list instance.

   ```
   device# route-map test permit 1 match ip address prefix-list pretest
   ```

5. Enter the **ip prefix-list** command in RBridge ID configuration mode to configure the IP prefix list instance.

```
device# config
device(config)# rbridge-id 1
device(config-rbridge-id-1)# ip prefix-list pretest permit 1.1.1.0/24
```

In the example above, when the **route-map test** *permit 1* command executes, only the static route 1.1.1.0/24 is exported into the OSPF domain, because there are no matching rules in **pretest** for route 11.11.11.0/24. The default action of **pretest** is **deny** (there is no match); therefore, the route 11.11.11.0/24 is not exported into the OSPF domain.

You can configure the router to permit or deny specific IP addresses explicitly. The router permits all IP addresses by default. If you want **permit** to remain the default behavior, define individual filters to deny specific IP addresses. If you want to change the default behavior to **deny**, define individual filters to permit specific IP addresses. Once you define a filter, the default action for addresses that do not match a filter is **deny**. To change the default action to **permit**, configure the last filter as **permit any any**.

IP over port-channel

Beginning with Network OS 7.0.0 and the introduction of IP Fabrics, support is provided over port-channels (Layer 2) for Layer 3 protocols.

Support for IP over port-channel provides the following advantages:

- Increases bandwidth between two RBridges.
- Provides fault tolerance, so that there is zero loss until the last member of the port-channel remains.
- Provides for dynamic bandwidth, through the removal or addition of port-channel members. (The upper layers do not need to know about the members, as these are device-dependent.)
- Provides load balancing.

This feature provides support for the following:

- Standard and Network OS port-channels, both static and dynamic
- IPv4 and IPv6 addressing
- Configuration and show commands as are currently supported for physical ports
- High availability

Support is provided for the following Layer 3 protocols:

- ARP/ND
- BFD
- BGP
- DHCP
- ICMP
- IGMP
- OSPFv2/v3
- PIM
- VRRP

In addition, support is provided for route-map policy.

Limitations

Note the following limitations:

- IP over vLAGs is not supported. If a port-channel with member interfaces from more than one node in logical chassis cluster mode, IPv4/IPv6 configurations are not allowed. If a port-channel (vLAG) with members from two nodes becomes segmented, it is no longer a vLAG operationally and IPv4/IPv6 configurations are not allowed.
- sFlow is not supported for Layer 3 port-channels.
- Layer 3 configurations, including IPv4/IPv6 address configurations, are not allowed on an "empty" port-channel (that is, one that has not yet been configured).
- A port-channel cannot be unconfigured until all Layer 3 configurations are removed from it.

Supported commands

The following commands, organized largely by protocol, support IP over port-channel.

IP routing commands

- **ip route**
- **ipv6 route**

Interface commands

- **clear ipv6 counters interface port-channel**
- **ip address**
- **ip address secondary**
- **ip mtu**
- **ipv6 address**
- **ipv6 address secondary**
- **ipv6 address use-link-local-only**
- **ipv6 address eui-64**
- **ipv6 address eui-64 secondary**
- **ipv6 address link-local**
- **ipv6 address anycast**
- **show ip interface**
- **show ipv6 interface**
- **show ip interface brief**
- **show ipv6 interface brief**
- **show ipv6 counters interface port-channel**
- **show port port-channel**
- **show port-channel**
- **show port-channel-redundancy-group**

ARP/ND commands

- **arp interface port-channel**
- **ip proxy-arp**
- **ip arp-aging-timeout**

- show arp port-channel
- show ipv6 neighbor port-channel

BGP commands

- neighbor

DHCP commands

- ip dhcp relay

ICMP commands

- ip icmp

IGMP commands

- ip igmp immediate-leave
- ip igmp last-member-query-count
- ip igmp last-member-query-interval
- ip igmp query-interval
- ip igmp query-max-response-time
- ip igmp robustness-variable
- ip igmp startup-query-count
- ip igmp startup-query-interval
- ip igmp static-group
- show ip igmp interface port-channel
- show ip igmp groups interface port-channel

OSPFv2 commands

- ip ospf active
- ip ospf area
- ip ospf auth-change-wait-time
- ip ospf autentication-key
- ip ospf bfd
- ip ospf cost
- ip ospf database-filter
- ip ospf dead-interval
- ip ospf hello-interval
- ip ospf md5-authentication
- ip ospf mtu-ignore
- ip ospf network
- ip ospf passive
- ip ospf priority
- ip ospf retransmit-interval
- ip ospf transmit-delay
- clear ip ospf counters port-channel
- show ip ospf interface port-channel

- show ip ospf neighbor port-channel
- show debug ip ospf internal interface port-channel

OSPFv3 commands

- ipv6 ospf active
- ipv6 ospf area
- ipv6 ospf autentication
- ipv6 ospf bfd
- ipv6 ospf cost
- ipv6 ospf dead-interval
- ipv6 ospf hello-interval
- ipv6 ospf instance
- ipv6 ospf mtu-ignore
- ipv6 ospf network
- ipv6 ospf passive
- ipv6 ospf priority
- ipv6 ospf retransmit-interval
- ipv6 ospf suppress-linklsa
- ipv6 ospf transmit-delay
- clear ipv6 ospf counts neighbor interface port-channel
- clear ipv6 ospf neighbor interface port-channel
- show ipv6 ospf interface port-channel
- show ipv6 ospf neighbor interface port-channel

PIM commands

- ip multicast-booundary
- ip pim-sparse
- ip pim dr-priority
- ip pim neighbor-filter
- show ip pim-sparse interface port-channel
- show ip pim neighbor interface port-channel

Route-map policy commands

- ip policy route-map
- ipv6 policy route-map
- match interface port-channel
- show route-map interface port-channel

VRRP commands

- clear vrrp statistics interface port-channel
- clear ipv6 vrrp statistics interface port-channel
- debug vrrp packets interface port-channel
- debug ipv6 vrrp packets interface port-channel

- **ipv6 vrrp-group**
- **show ipv6 vrrp interface port-channel**
- **show vrrp interface port-channel**
- **vrrp-group**

IP unnumbered interfaces

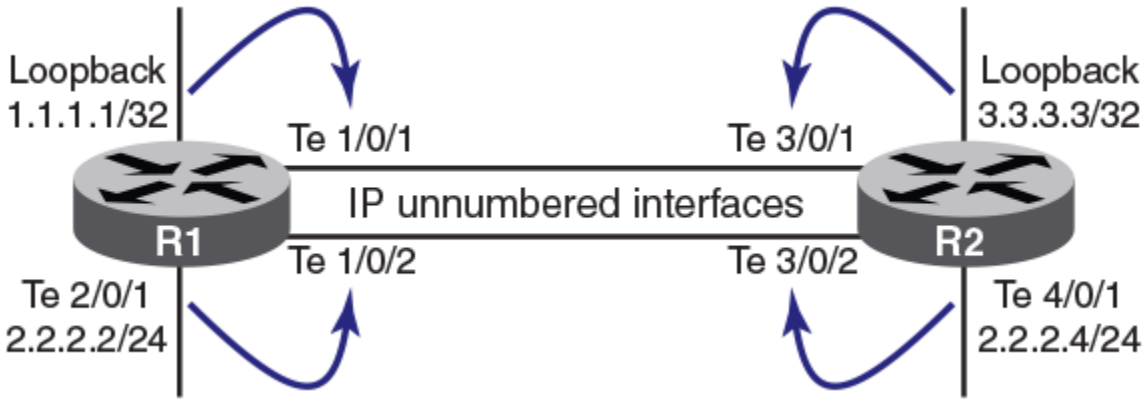
An IP unnumbered interface can borrow an IP address from another Layer 3 interface that is already configured on the device.

This address is used as a source address in the Layer 3 packets that are sent out of the IP unnumbered interface. The interface from which the IPv4 address is borrowed is referred to as the "donor" interface. The donor interface can be any other Layer 3 interface—physical, virtual Ethernet (VE), loopback, and so on.

When numbered IP interfaces are used, each interface is assigned an IPv4 address from a unique subnet. This means a router cannot have overlapping connected routes. Typically, when enabling Layer 3 over directly connected routers you must assign an IP address to both the routers, which are connected to same interface and belong to the same subnet. Therefore, the total number of IP addresses used depends on the number of Layer 3 interfaces supported by the router. With a large number of Layer 3 interfaces available, this scheme results in a waste of IP addresses and network subnets. The /31 support reduces consumption of addresses; however, two IP addresses are consumed per interface. The IP unnumbered interfaces feature reduces network and address space consumption for supporting Layer 3 over router interfaces.

In the following diagram, interfaces Te 1/0/1, Te 1/0/2, Te3/0/1, and Te 3/0/2 are IP unnumbered interfaces borrowing IPv4 addresses from either loopback interface or another physical interface. Both endpoints of the unnumbered interface may or may not be in same subnet.

FIGURE 2 IP unnumbered interfaces



IP unnumbered interfaces guidelines and restrictions

Follow these guidelines and restrictions when configuring IP unnumbered interfaces.

- You can have multiple donor interfaces in the device, with each donor interface having multiple IP unnumbered interfaces inherit its IP address. However, an IP unnumbered interface can have only one donor interface.
- If the donor interface is down (as a result of a link state or administrative state), a ping to the donor IP address fails, even if the IP unnumbered interfaces that inherited the IP address are available.

- LLDP is used to detect neighbors attached to an IP unnumbered interface. As a prerequisite, LLDP must be enabled.
- Only physical interfaces and Layer 3 port-channels can be configured as IP unnumbered interfaces. IP unnumbered interfaces are not supported for VE or switch virtual interfaces (SVIs).
- IP unnumbered interface support is available for the IPv4 address family only. Once the interface is configured as an IP unnumbered interface, it is treated as a point-to-point interface, so there can be only one remote neighbor attached to this interface.
- IP address configurations are the only configurations that the IP unnumbered interfaces can inherit from the donor interface. All other configurations (such as ICMP, ACLs, DHCP, and PBR) that are configured on the donor interface apply only to the donor interface and are not inherited by the IP unnumbered interfaces. You must configure these features separately on the IP unnumbered interfaces.
- The IP unnumbered interfaces feature is supported on the default VRF and user-defined VRFs. Both the donor and the IP unnumbered interfaces must be in the same VRF. IP unnumbered interfaces are not supported in the management VRF.
- Because there is no network subnet associated with the unnumbered IP interface, ARP is not supported on unnumbered IP interfaces.
- IP unnumbered interfaces do not support VRRP/VRRP-E.
- IP unnumbered interfaces do not support multicast.

NOTE
When eBGP is used in conjunction with the IP unnumbered interfaces feature, it is important to ensure that sufficient Time To Live (TTL) values are available for hello messages to establish separate neighbor adjacencies on leaf switches in an IP Fabric. To do so, use the **neighbor ebgp-multihop** command and set the number of maximum hops to **2**.

Configuring an unnumbered IP interface

Use these commands to configure a standard IPv4 interface as an unnumbered IP interface.

1. Enter global configuration mode.

```
device# configure terminal
```

2. Specify an Ethernet interface.

```
device(config)# interface TenGigabitEthernet 1/0/2
```

3. Use the **ip unnumbered** command to set the IP interface to be unnumbered and specify a loopback interface as the address donor. This command can be entered multiple times, if required.

```
device(config-if-te-1/0/2)# ip unnumbered loopback 2
```

NOTE
You must remove this configuration before you can specify a numbered interface.

4. Use the **show ip interface brief** command to confirm the configuration.

```
device(config)#do show ip interface brief

Flags:  U - Unnumbered interface
Interface                     IP-Address      Vrf                 Status                Protocol
==========================    ==========      ==================  ====================  ========
FortyGigabitEthernet 1/0/49    unassigned        default-vrf       up                    up
FortyGigabitEthernet 1/0/50    unassigned        default-vrf       up                    up
FortyGigabitEthernet 1/0/51    unassigned        default-vrf       up                    up
FortyGigabitEthernet 1/0/52    unassigned        default-vrf       up                    up
TenGigabitEthernet 1/0/1       1.1.1.1           default-vrf       up                    up
TenGigabitEthernet 1/0/2       1.1.1.1 (U)       default-vrf       up                    up
TenGigabitEthernet 1/0/3       unassigned        default-vrf       up                    up
```

You can use the **show ip route** command to confirm neighbor routes.

ECMP with IP unnumbered

The ECMP with IP unnumbered feature provides automatically assigned metrics for BGP neighbor addresses that are learned through interfaces configured as IP unnumbered.

In releases prior to Network OS 7.2.0, attached interfaces were considered to be "equal" with respect to Layer 3 Equal-Cost Multipath (ECMP), even if not all port speeds were the same. Such speed differences can introduce problems, including dropped traffic. For example, if two devices are connected by means 1-G, 10-G and 40-G ports, all of the routes learned on the IP unnumbered interfaces are considered equal (under ECMP) with respect to the IP route table. This feature allows you to create primary interfaces (of the same speed), as well as secondary interfaces that may be of a different speed and therefore would produce a different metric in Layer 3 ECMP. With this feature, the default behavior causes the metric to be calculated according to the interface link speed. No user configuration is required.

In case of a downgrade to a release prior to Network OS 7.2.0, all BGP auto neighbor discovery configurations must be removed.

DNS

The Domain Name System (DNS) operates as a hierarchical naming system that assigns a name (such as a company name) to an Internet entity to represent the real IP address of the entity. An entity can be a gateway router and is referred to as a domain.

A domain name (for example, extreme.router.com) can be used in place of an IP address for certain operations such as IP pings, trace routes, and Telnet management connections to the router. Instead of having to remember all the numbers of the IP address, a domain name is easier to remember. DNS is comprised of the following:

- DNS Server
- DNS Resolver
- DNS gateway addresses

DNS Server

A DNS server stores the information about a DNS domain. DNS servers are a key element of DNS because they respond to queries against its database. When a DNS domain is defined on this device to recognize all hosts within that domain, this device automatically appends the appropriate domain to the host address and forwards it to the domain name server.

DNS Resolver

The DNS resolver is a feature in a Layer 2 or Layer 3 device that sends and receives queries to and from the DNS server on behalf of a client. You can create a list of domain names that can be used to resolve host names. This list can have more than one domain name. When a client performs a DNS query, all hosts within the domains in the list can be recognized and queries can be sent to any domain on the list. After you define a domain name, the device automatically appends the appropriate domain to a host and forwards it to the DNS servers for resolution.

DNS Gateway Addresses

Gateway IP addresses assigned to the device through enable clients attached to the device to reach DNS.

Configuring DNS

A Domain Name System (DNS) domain and DNS gateway addresses can be configured to resolve host names to IP addresses.

1. Enter global configuration mode.

```
device# configure terminal
```

2. Use the **ip dns domain-name** to configure a domain name.

```
device(config)# ip dns domain-name mycompany.com
```

3. Use the **exit** command to return to privileged EXEC mode.

```
device(config)# exit
```

4. (Optional) Use the **traceroute** command to verify the DNS configuration.

```
device# traceroute mycompany.com

Sending DNS Query to 10.157.22.199
Tracing Route to IP node 10.157.22.80
To ABORT Trace Route, Please use stop-traceroute command.
 Traced route to target IP node 10.157.22.80:
  IP Address    Round Trip Time1    Round Trip Time2
  10.95.6.30    93 msec             121 msec
```

   The output shows that 10.157.22.199 is the IP address of the domain name server (default DNS gateway address), and 10.157.22.80 represents the IP address of the mycompany.com host.

The following example configures a DNS domain; and default and secondary DNS gateway addresses for DNS servers

```
device# configure terminal
device(config)# ip dns domain-name www.mycompany.com
device(config)# ip dns name-server 10.157.22.199
device(config)# exit
device(config)# ip dns name-server 10.96.7.15
```

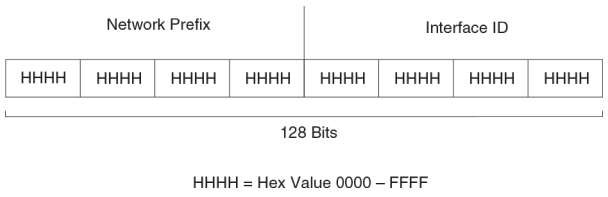
IPv6 addressing

IPv6 addressing overview

IPv6 increases the number of network address bits from 32 (IPv4) to 128 bits, which provides more unique IP addresses to support increasing number of network devices.

An IPv6 address comprise 8 fields of 16-bit hexadecimal values separated by colons (:). The following figure shows the IPv6 address format.

FIGURE 3 IPv6 address format



As shown in the above figure, 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 this IPv6 address includes hexadecimal fields of zeros. To make the address manageable, you can:

- 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, consider the following:

- You can use the two colons (::) only once in the address to represent the longest successive hexadecimal fields of zeros.
- The hexadecimal letters in IPv6 addresses are not case-sensitive.

As shown in Figure 3, 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 network portion of the IPV6 address.

The following is an example of an IPv6 prefix.

2001:DB8:49EA:D088::/64

Understanding IPv6 addresses and prefixes

To forward IPv6 traffic on an interface, the interface must have an IPv6 address. IPv6 is enabled globally by default. By default, an IPv6 address is not configured on an interface. When you configure a global 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 assign a link-local IPv6 address to the interface, the address is automatically computed for the interface. If preferred, you can override the automatically configured link-local address with an address that you manually configure.

Configuring a global IPv6 address on an interface does the following:

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

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

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

The IPv6 Neighbor Discovery feature sends messages to these multicast groups.

The representation of IPv6 address prefixes is similar to that for IPv4 address prefixes written in Classless Inter-Domain Routing (CIDR) notation. An IPv6 address prefix is represented as *ipv6-prefix/prefix-length*, where *ipv6-prefix* is an IPv6 address in any of the notations listed in RFC 4291, and *prefix-length* is a decimal value specifying how many of the left-most contiguous bits of the address comprise the prefix.

For example, the following are legal representations of the 60-bit prefix 20010DB80000CD3 (hexadecimal):

- 2001:0DB8:0000:CD30:0000:0000:0000:0000/60
- 2001:0DB8::CD30:0:0:0:0/60
- 2001:0DB8:0:CD30::/60

Network OS supports IPv6 prefixes through the **ipv6 address** command and its various options. You must specify the *ipv6-address* parameter in hexadecimal by using 16-bit values between colons as documented in RFC 4291. You must specify the *prefix-length* parameter as a decimal value. A slash mark (/) must follow the *ipv6-address* parameter and precede the *prefix-length* parameter.

Note the following options:

- The **eui-64** keyword configures the global address with an EUI-64 interface ID in the low-order 64 bits. The interface ID is automatically constructed in IEEE EUI-64 format by using the MAC address of the interface. 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.
- The optional **secondary** keyword specifies the address as a secondary address.
- The optional **anycast** keyword configures an anycast address for a set of interfaces that belong to different nodes

Understanding dual-stack support

It is not expected that practical IPv6 implementations will be made without consideration for existing IPv4 networks. Making a transition to IPv6 networks without significant challenges or service disruptions is an incremental, step-by-step process. Interoperability testing must focus, where applicable, on such areas as firewalls, voice service, wireless service, and application-layer interfaces between popular applications and implementation of the IPv6 protocol stack. Care must be taken to test server-side and client-side interoperability not only in pure IPv6 configurations, but also mixed IPv4 and IPv6 configurations, across multiple topologies and vendor platforms.

> NOTE
> Support for IPv6 is provided on the VDX 6740 series, VDX 6940 series, and the VDX 8770 series. IPv6 functionality is enabled by default.

In a data-center network design, depending on the functionality required, switches at both the access layer (if there are no service appliances, such as firewalls, load balancers, and so on at the aggregation layer) and the aggregation layer (if there are appliances hanging off aggregations switches) require IPv6 routing and IPv6/IPv4 termination, as summarized below.

FIGURE 4 IPv6 dual-stack network

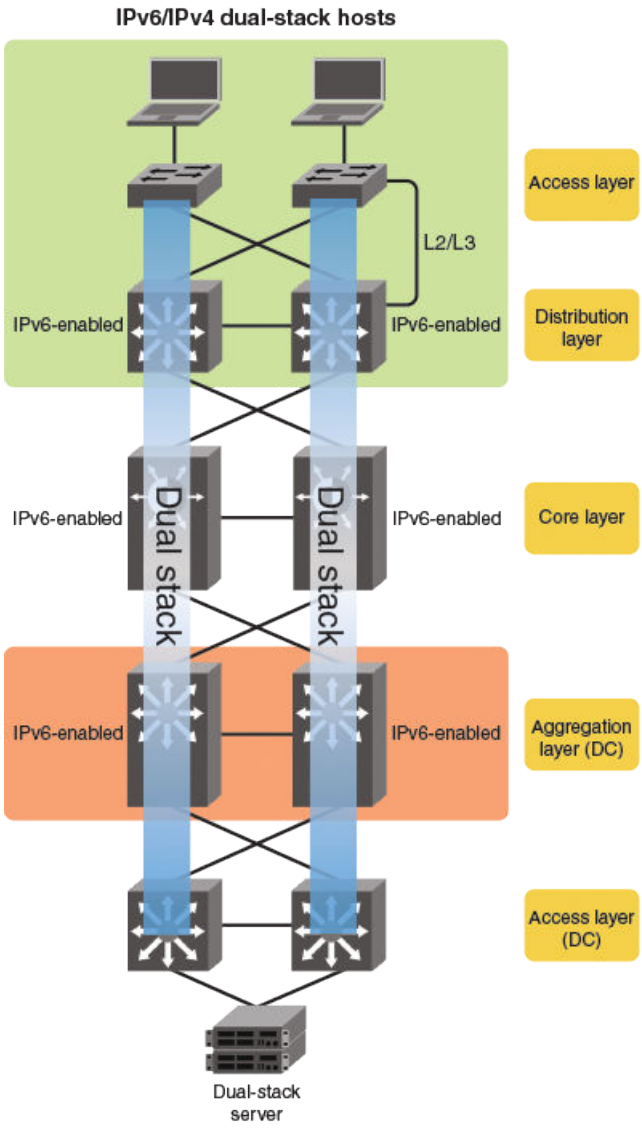


TABLE 7 Switch requirements to support dual-stack implementations at the access and aggregation layers

Access layer	Aggregation layer
Access switches must provide IPv6/IPv4 termination boundary and IPv6 among switches in east-west and north-south directions.	Aggregation switches must support the following: • Policy-Based Routing (PBR) • IPv6-based routing • IPv6/IPv4 termination (if services are based on IPv4)
Peer aggregation-layer switches must support IPv6-based routing.	

This document does not attempt to provide a detailed discussion of the well-known IPv6 protocol. For details, refer to the following RFCs for the appropriate technology areas.

TABLE 8 IPv6 RFCs

Technology	RFC	Description
IPv6 core	6434	IPv6 Node Requirements (a listing of all major RFCs related to IPv6
	2460	IPv6 Specification
	4861/5942	IPv6 Neighbor Discovery
	2462	IPv6 Stateless Address Auto-Configuration
	4443	ICMPv6 (replaces RFC 2463)
	4291	IPv6 Addressing Architecture
	3587	IPv6 Global Unicast Address Format
	2375	IPv6 Multicast Address Assignments
	2464	Transmission of IPv6 over Ethernet Networks
	2711	IPv6 Router Alert Option
	3596	DNS Support
IPv6 routing	2740	OSPFv3 for IPv6
	2545	Use of BGP-MP Extensions for IPv6
IPv6 multicast	2710	Multicast Listener Discovery (MLD) for IPv6
	3810	Multicast LIstener Discovery Version 2 for IPv6
	4604	IGMPv3 and MLDv2 for SSM
	4607	Source-Specific Multicast for IP
	4601	PIM-SM
IPv6 transitioning	2893	Transition Mechanisms for IPv6 Hosts and Routers
	3056	Connection of IPv6 Domains via IPv4 Clouds
Network management	3176	sFlow
	5798	VRRP Version 3 for IPv4 and IPv6

Configuring a global IPv6 address with a manually configured interface ID

To configure a global IPv6 address for an interface, including a manually configured interface ID, do the following.

1. In global configuration mode, select an interface.

```
switch(config)# interface te 3/1/1
```

2. Enter the **ipv6 address** *ipv6-prefix/prefix-length* command in interface subtype configuration mode, as in the following example.

```
switch(config-if-te-3/1/1)# ipv6 address 2001:200:12d:1300:240:d0ff:
fe48:4672/64
```

3. Optionally, you can use the **secondary** keyword with this command to configure one or more secondary addresses, with the following restrictions:
   - You can configure a maximum of 256 secondary addresses.
   - You cannot configure a secondary address on an interface without first configuring a primary address.
   - You cannot delete the primary address (by means of the **no ipv6 address** command) without first deleting all secondary addresses.
   - Secondary addresses are not supported on loopback or management interfaces.

Configuring a global IPv6 address with an automatically computed EUI-64 interface ID

To configure a global IPv6 address with an automatically computed EUI-64 interface ID in the low-order 64 bits, do the following.

1. In global configuration mode, select an interface.

```
switch(config)# interface te 3/1/1
```

2. Enter the **ipv6 address** *ipv6-prefix/prefix-length* **eui-64** command in interface subtype configuration mode, as in the following example.

```
switch(config-if-te-3/1/1)# ipv6 address 2001:db8:12d:1300::/64 eui-64
```

   NOTE
   The **eui--64** keyword configures the global address with an EUI-64 interface ID in the low-order 64 bits. The interface ID is automatically constructed in IEEE EUI-64 format by means of the interface's MAC address.

3. Optionally, you can use the **secondary** keyword with this command to configure one or more secondary addresses, with the following restrictions:
   - You can configure a maximum of 256 secondary addresses.
   - You cannot configure a secondary address on an interface without first configuring a primary address.
   - You cannot delete the primary address (by means of the **no ipv6 address** command) without first deleting all secondary addresses.
   - Secondary addresses are not supported on loopback or management interfaces.

Configuring a link-local IPv6 address

To configure a link-local IPv6 address without configuring a global or site-local address for the interface, do the following.

1. In global configuration mode, select an interface.

```
switch(config)# interface te 3/1/1
```

2. Enable IPv6 on the interface. You can configure an automatically computed address (by using the **ipv6 address use-link-local-only** command), or an explicit address (by using the **ipv6 address link-local** command. The latter command overrides the automatically computed address.
   - The following configures an automatically computed address:

     ```
     switch(config-if-te-3/1/1)# ipv6 address use-link-local-only
     ```

   - The following configures an explicit address:

     ```
     switch(config-if-te-3/1/1)# ipv6 address fe80::240:d0ff:fe48:4672 link-local
     ```

     NOTE
     When configuring VLANs that share a common tagged interface with a virtual Ethernet (VE) interface, it is recommended 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.

Configuring an IPv6 anycast address

In IPv6, an anycast address is an address for a set of interfaces that belong to different nodes. Sending a packet to an anycast address results in the delivery of the packet to the closest interface that has an 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 device, you configure an interface assigned an anycast address to recognize the address as an anycast address.

NOTE
Duplicate address detection (DAD) is not supported for anycast addresses.

To configure an anycast address on an interface, do the following.

1. In global configuration mode, select an interface.

   ```
   switch(config)# interface te 3/1/1
   ```

2. Enter the **ipv6 address** *ipv6-prefix/prefix-length* **anycast** command in interface subtype configuration mode, as in the following example.

   ```
   switch(config-if-te-3/1/1)# ipv6 address 2002::6/64 anycast
   ```

Configuring IPv4 and IPv6 protocol stacks

If a device is deployed as an endpoint for an IPv6 over IPv4 tunnel, you must configure the device to support IPv4 and IPv6 protocol stacks. Each interface that sends and receives IPv4 and IPv6 traffic must be configured with an IPv4 address and an IPv6 address.

Do the following to configure an interface to support both IPv4 and IPv6 protocol stacks.

1. Select an interface to support dual stacks.

   ```
   switch(config)# interface te 3/1/1
   ```

2. Assign an IPv4 address and mask to the interface.

```
switch(config-if-te-3/1/1)# ip address 192.168.1.1 255.255.255.0
```

3. Assign an IPv6 address with an automatically computed EUI-64 interface ID.

```
switch(config-if-te-3/1/1)# ipv6 address 2001:200:12d:1300::/64 eui-64
```

Configuring an IPv6 address family

You can enable IPv6 address-family support for VRF unicast routing, by means of the **address-family ipv6 unicast** command. This allows you to do the following:

- Configure IPv6 static routes in the specified VRF
- Configure IPv6 static neighbors in the specified VRF
- Configure any per-VRF routing-protocol options

Do the following to configure an IPv6 address family.

1. In RBridge ID configuration mode, create a VRF instance.

```
switch(config-rbridge-id-1)# vrf red
```

2. In VRF configuration mode, enable IPv6 address-family unicast support.

```
switch(config-vrf-red)# address-family ipv6 unicast
```

3. Specify an interface.

```
switch(config)# int te 1/0/10
```

4. In interface subtype configuration mode, enable VRF forwarding and specify an IPv6 address and prefix length.

```
switch(conf-if-te-1/0/10)# vrf forwarding red
switch(conf-if-te-1/0/10)# ipv6 address 1111::1111/64
```

If IPv6 address-family unicast support is not enabled, an error is returned as in the following example.

```
switch(conf-if-te-1/0/10)# rb 1
switch(config-rbridge-id-1)# vrf blue
switch(config-vrf-blue)# int te 1/0/10
switch(conf-if-te-1/0/10)# vrf forwarding blue
switch(conf-if-te-1/0/10)# ipv6 address 1111::2222/64
%% Error: VRF Address Family not configured
```

Configuring static IPv6 routes

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

You must first enable IPv6 on at least one interface in RBridge ID configuration mode by configuring an IPv6 address or explicitly enabling IPv6 on that interface. Static routes are VRF-specific; they implicitly use the default VRF unless a nondefault VRF is configured (as in the last of the following examples).

The following tasks illustrate a variety of static IPv6 route configurations.

To configure a static IPv6 route with a destination and next-hop gateway address as a global unicast interface, perform the following task in global configuration mode.

1. Enter RBridge ID configuration mode.

```
switch(config)# rbridge-id 54
```

2. Configure a static IPv6 route with a destination and next-hop gateway address as a global unicast interface.

```
switch(rbridge-id-54)# ipv6 route 3ffe:abcd::/64 2001::0011:1234
```

To configure a static IPv6 route with a destination and next-hop gateway address as a global unicast interface with a metric of 10 and an administrative distance of 110, perform the following task in global configuration mode.

1. Enter RBridge ID configuration mode.

```
switch(config)# rbridge-id 54
```

2. Configure a static IPv6 route with a destination and next-hop gateway address as a global unicast interface with a metric of 10 and an administrative distance of 110.

```
switch(rbridge-id-54)# ipv6 route 3ffe:abcd::/64 2001::0011:1234 10 distance 110
```

NOTE

The value specified for *metric* is used by the Layer 3 switch to compare 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 Layer 3 switch has already placed in the IPv6 static route table. Two or more routes to the same destination with the same metric will load share (as in ECMP load sharing). Values range from 1 through 16 with a default of 1.

The value specified by **distance** is used by the Layer 3 switch to compare this route with routes from other route sources that have the same destination. By default, static routes take precedence over routes learned by routing protocols. To choose a dynamic route over a static route, configure the static route with a higher administrative distance than the dynamic route. The range for *number* is from 1 through 255, with a default of 1.

To configure a static IPv6 route with a destination global unicast address and next-hop gateway address on a tengigabit Ethernet interface, perform the following task in global configuration mode.

1. Enter RBridge ID configuration mode.

```
switch(config)# rbridge-id 54
```

2. Configure a static IPv6 route with a destination global unicast address and next-hop gateway address.

```
switch(rbridge-id-54)# ipv6 route 2001:db8::0/32 2001:db8:0:ee44::1
```

To configure a static IPv6 route with a destination and next-hop gateway address as a link-local address on an Ethernet interface, perform the following task in global configuration mode.

1. Enter RBridge ID configuration mode.

```
switch(config)# rbridge-id 54
```

2. Configure a static IPv6 route with a destination and next-hop gateway address as a link-local address on an Ethernet interface.

```
switch(rbridge-id-54)# ipv6 route 3001:1234::/64 fe80::1234
```

To configure a static IPv6 route with a destination and next-hop gateway address and cause packets to those addresses to be dropped by shunting them to the "null0" interface, perform the following task in global configuration mode.

1. Enter RBridge ID configuration mode.

```
switch(config)# rbridge-id 54
```

2. Configure a static IPv6 route with a destination and next-hop gateway address and cause packets to those addresses to be dropped by shunting them to the "null0" interface.

```
switch(rbridge-id-54)# ipv6 route 2fe0:1234:5678::/64 null 0
```

NOTE
This is called "black-holing." It is recommend that this option be used to block undesirable traffic or traffic generated in DoS attacks. Traffic is routed dynamically to a "dead" interface. It can also be directed to a host so that information can be collected for investigation.

To configure a static route and next-hop gateway on a virtual Ethernet interface, perform the following task in global configuration mode.

1. Enter RBridge ID configuration mode.

```
switch(config)# rbridge-id 54
```

2. Configure a static route and next-hop gateway on a virtual Ethernet interface.

```
switch(rbridge-id-54)# ipv6 route 2fe0:1234:5678::/64 ve 500
```

To configure a static route in a nondefault VRF, perform the following task in global configuration mode.

1. Enter RBridge ID configuration mode.

```
switch(config)# rbridge-id 54
```

2. Create a nondefault VRF instance, by using the **vrf** command.

```
switch(rbridge-id-54)# vrf network_70
```

3. Enter IPv6 address-family configuration mode, by using the **address-family ipv6 unicast** command.

```
switch(config-vrf-network_70)# address-family ipv6 unicast
```

4. Configure a static route.

```
switch(vrf-ipv6-unicast)# ipv6 route 2000:1000::/64 3100:1000::1245
```

NOTE
All other static route configurations, either IPv6 or IPv4, are valid under address-family configuration mode for nondefault VRFs.

For details of the **show** and **clear** commands for IPv6 routes, refer to Monitoring and managing IPv6 networks on page 67.

Changing the IPv6 MTU

The IPv6 MTU is the maximum length in bytes of an IPv6 packet that can be transmitted on a particular interface. If an IPv6 packet is longer than the defined MTU, the originating host breaks the packet into fragments that are shorter than that MTU. Per RFC 2460, the minimum IPv6 MTU for any interface is 1280 bytes; the default maximum is 1500 bytes. You can change the MTU both at the interface level and globally.

NOTE
If the size of a jumbo frame received on a port is equal to the maximum frame size and this value is greater than the IPv4/IPv6 MTU of the outgoing port, the jumbo frame is forwarded to the CPU.

An IPv6 interface can obtain an IPv6 MTU value from any of the following sources:

- Default IPv6 MTU setting

- Global IPv6 MTU setting
- Interface IPv6 MTU setting

An interface determines the actual MTU value as follows:

- If an IPv6 interface MTU value is configured, the configured value is used.
- If an IPv6 interface value is not configured and N IPv6 global MTU value is configured, the configured global value is used.
- If neither an IPv6 interface value nor an IPv6 global MTU value is configured, the default IPv6 MTU value of 1500 is used.

You can specify between [1284 - (*default-max-frame-size*) - 18]. If a nondefault value is configured for an interface, router advertisements include an MTU option. The minimum value you can configure is 1298 * (IP6_MIN_MTU + 18 for Ethernet ports).

To change the IPv6 MTU on an interface, use the **ipv6 mtu** command as in the following example.

```
switch(config)# int te 3/0/1
switch(config-if-te-3/0/1)# ipv6 mtu 1280
```

ATTENTION

To route packets larger than 2500 bytes (the default for an Ethernet interface), you must also use the **mtu** command to set the same MTU value on the interface as that set by the **ipv6 mtu** command. Otherwise packets will be dropped . The range for the **mtu** command is from 1522 through 9219 bytes.

Configuring IPv6 Neighbor Discovery

The Neighbor Discovery feature for IPv6 uses ICMPv6 messages to do the following:

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

An IPv6 host is required to listen for and recognize the following addresses, which identify this host:

- 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 (NS) messages for duplicate address detection (DAD)
- Router Advertisement (RA) messages:
  - The interval between RA messages
  - A lifetime value that indicates a router is advertised as a default router (for use by all nodes on a given link)
  - Prefixes advertised in RA messages
  - Flags for host stateful autoconfiguration
- The time that an IPv6 node considers a remote node to be reachable (for use by all nodes on a given link)
- NS interval for reachability

Neighbor Discovery configurations are executed at the interface level, by means of a series of **ipv6 nd** commands.

Neighbor Solicitation and Neighbor Advertisement messages

Neighbor Solicitation (NS) and Neighbor Advertisement (NA) 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 NS message. The NS 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

  NOTE
  The source address is unspecified for DAD.

- **Destination address:** Solicited-node multicast address (FF02:0:0:0:0:1:FF00::/104) that corresponds the IPv6 address of node 2

  NOTE
  For unicast NS traffic, the destination address is the IPv6 address of the destination. For non-unicast NS traffic, the destination address is the solicited-node's multicast address.

- Link-layer address of node 1
- A query for the link-layer address of node 2

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

- **Source address:** IPv6 address of the node 2 interface that sends the message

  NOTE
  For solicited advertisements, this is the source address of an invoking NS. Alternatively, if the source address of the NS is the unspecified address, this is the all-nodes multicast address.

- **Destination address:** IPv6 address of node 1
- Link-layer address of node 2

After node 1 receives the NA 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 NS 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 NA message to the all-nodes multicast address (FF02::1) if there is a change in their link-layer address.

Router Advertisement and Router Solicitation messages

Router Advertisement (RA) and Router Solicitation (RS) messages enable a node on a link to discover the routers on the same link.

RA messages are sent periodically by a router to provide the following information to hosts:

- Router information such as link-layer address and lifetime of the prefix
- IPv6 prefixes for address autoconfiguration
- Network information such as MTU and hop limit
- Additional information such as reachable time, retransmission time for neighbor solicitations

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

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

At system startup, a host on a link sends an RS message to the all-routers multicast address (FF02::2). Sending an RS message, which has a value of 133 in the Type field of the ICMP packet header, immediately enables the host to configure its IPv6 address automatically, instead of having to wait for the next periodic RA message.

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

You can configure a variety of RA message parameters at the interface level.

Neighbor Redirect messages

After forwarding a packet, by default a router can send a Neighbor Redirect (NR) message to a host to inform it of a "better" first-hop router. The host receiving the NR message will then readdress the packet to the better router.

A device sends an NR message only for unicast packets, only to the originating node, and to be processed by the node.

An NR message has a value of 137 in the Type field of the ICMP packet header.

Duplicate address detection (DAD)

IPv4 nodes use ARP Request messages and a method called *gratuitous ARP* to detect a duplicate unicast IPv4 address on the local link. Similarly, IPv6 nodes use Neighbor Solicitation messages to detect the use of duplicate addresses on the local link in a process known as *duplicate address detection (DAD)*, as described in RFC 4862.

With IPv4 gratuitous ARP, the Source Protocol Address and Target Protocol Address fields in the ARP Request message header are set to the IPv4 address for which duplication is being detected. In IPv6 DAD, the Target Address field in the Neighbor Solicitation (NS) message is set to the IPv6 address for which duplication is being detected. DAD differs from address resolution in the following ways:

- In the DAD NS message, the Source Address field in the IPv6 header is set to the unspecified address (::). The address being queried for duplication cannot be used until it is determined that there are no duplicates.
- In the Neighbor Advertisement (NA) reply to a DAD NS message, the Destination Address in the IPv6 header is set to the link-local all-nodes multicast address (FF02::1). The Solicited flag in the NA message is set to 0. Because the sender of the DAD NS message is not using the desired IP address, it cannot receive unicast NA messages. Therefore, the NA message is multicast.
- Upon receipt of the multicast NA message with the Target Address field set to the IP address for which duplication is being detected, the node disables the use of the duplicate IP address on the interface. If the node does not receive an NA message that defends the use of the address, it initializes the address on the interface.

An IPv6 node does not perform DAD for anycast addresses. Anycast addresses are not unique to a node. Network OS does not perform DAD for IPv6 addresses configured on loopback interfaces.

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

If DAD 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.

Setting Neighbor Solicitation parameters for DAD

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

If DAD 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 (NS) message parameters that affect DAD while it verifies that a tentative unicast IPv6 address is unique:

- The number of consecutive NS messages that DAD sends on an interface. By default, DAD sends two NS messages without any follow-up messages.
- The interval in seconds at which DAD sends a NS message on an interface. By default, DAD sends an NS message every 1 second.

  NOTE
  For the interval at which DAD sends an NS message on an interface, the router uses seconds as the unit of measure instead of milliseconds.

For example, to change the number of NS messages sent on Ethernet interface 3/1/1 to 3 and the interval between the transmission of the two messages to 4 seconds, do the following

1. In global configuration mode, select an interface.

   ```
   switch(config)# interface te 3/1/1
   ```

2. Enter the **ipv6 nd dad attempt** *number* command to set the number of solicitation messages sent on the interface.

   ```
   switch(config-if-te-3/1/1)# ipv6 nd dad attempt 3
   ```

   NOTE
   To disable DAD on an interface, set the number of attempts to 0.

3. Enter the **ipv6 nd dad time** *seconds* command to set the interval between the messages.

   ```
   switch(config-if-te-3/1/1)# ipv6 nd dad time 4
   ```

   NOTE
   It is recommended that you do not specify intervals that are very short in normal IPv6 operation. When a nondefault interval value is configured, that interval is both advertised and used by the router itself.

Configuring IPv6 static neighbor entries

In some cases a neighbor cannot be reached by means of Neighbor Discovery. To resolve this you can add a static entry to the ND cache, causing a neighbor to reachable at all times. (A static IPv6 ND entry is like a static IPv4 ARP entry.)

For example, use the **ipv6 neighbor** command in interface subtype configuration mode to add a static entry for a neighbor with IPv6 address 2001:db8:2678::2 and link-layer address 0000.002b.8641, reachable through interface te 3/0/1.

```
switch(config-if-te-3/0/1)# ipv6 neighbor 2001:db8:2678::2 0000.002b.8641
```

Setting IPv6 Router Advertisement parameters

You can adjust the following parameters for Router Advertisement (RA) messages:

- The interval (in seconds) at which an interface sends RA messages. By default, an interface sends an RA message randomly, every 200 to 600 seconds.
- The "router lifetime" value, which is included in RA messages sent from a particular interface. The value (in seconds) indicates whether 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.

  NOTE
  When adjusting these parameter settings, it is recommended that you set the interval between router advertisement transmission to 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 specify the range matching the configuration and the router lifetime value to 1900 seconds on an interface, enter the following commands.

1. In global configuration mode, select an interface.

   ```
   switch(config)# interface te 3/1/1
   ```

2. Enter the **ipv6 nd ra-interval** command to set a maximum interval range and minimum interval at which RA messages are sent.

   ```
   switch(config-if-te-3/1/1)# ipv6 nd ra-interval 1200 min 400
   ```

3. Enter the **ipv6 nd ra-lifetime** *number* command to set the RA message lifetime.

   ```
   switch(config-if-te-3/1/1)# ipv6 nd ra-lifetime 1900
   ```

4. Enter the **ipv6 nd hoplimit** command to specify a nondefault hop limit.

   ```
   switch(config-if-te-3/1/1)# ipv6 nd hoplimit 32
   ```

5. Enter the **ipv6 nd mtu** command to specify a nondefault MTU.

   ```
   switch(config-if-te-3/1/1)# ipv6 nd mtu 2400
   ```

Controlling prefixes advertised in IPv6 Router Advertisement messages

By default, Router Advertisement (RA) messages include prefixes configured as addresses on interfaces by means of the **ipv6 address** command. You can use the **ipv6 nd prefix** command to control exactly which prefixes are included in RA messages. The prefix is associated with a valid, preferred lifetime. For a prefix derived from the global address, the lifetime value is infinite (0xFFFFFFFF).

RA messages also use 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 autoconfiguration feature can use the specified prefix in the automatic configuration of 128-bit IPv6 addresses for hosts on the local link.

The following illustrates the execution of the **ipv6 nd prefix** command.

1. In global configuration mode, select an interface.

```
switch(config)# interface te 3/1/1
```

2. Enter the **ipv6 nd prefix**command and specify a prefix and prefix length.

```
switch(config-if-te-3/1/1)# ipv6 nd prefix 2ffe:1111::/64
```

NOTE
Valid and preferred lifetimes are default values, which are 2592000 and 604800, respectively.

Setting flags in IPv6 Router Advertisement messages

An IPv6 Router Advertisement (RA) message can include the following flags:

- **Managed Address Configuration** -- This flag indicates to hosts on a local link whether 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 whether the hosts can get information that is not address-related 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 whether they can autoconfiguration information that is not address-related. If the flag is set, the hosts can use stateful autoconfiguration to get non-IPv6-address information.

NOTE
When determining whether hosts can use stateful autoconfiguration to get non-IPv6-address information, use a Managed Address Configuration flag to override an unset Other Stateful Configuration flag. In this situation, the hosts can obtain non-IPv6-address 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 RA messages. For example, to set these flags in router advertisement messages sent from an interface, enter the following commands.

1. In global configuration mode, select an interface.

```
switch(config)# interface te 3/1/1
```

2. Enter the **ipv6 nd managed-config-flag**command to enable hosts on a local link to use stateful autoconfiguration to get addresses as well as non-IPv6-address information.

```
switch(config-if-te-3/1/1)# ipv6 nd managed-config-flag
```

3. Enter the **ipv6 nd other-config-flag**command to enable hosts on a local link to use stateful autoconfiguration non-IPv6-address information.

```
switch(config-if-te-3/1/1)# ipv6 nd other-config-flag
```

Monitoring and managing IPv6 networks

You can monitor and manage IPv6 networks by using a variety of **show** and **clear** commands. For command details, refer to the *Extreme Network OS Command Reference*.

The following table lists the available **show** commands for IPv6 networks.

TABLE 9 IPv6 show commands

Command	Description
show ipv6 counters interface	Displays the counters on an IPv6 interface.
show ipv6 interface	Displays details of IPv6 interfaces.
show ipv6 nd interface	Displays information about the IPv6 Neighbor Discovery configuration on an interface
show ipv6 route	Displays information about IPv6 routes.
show ipv6 static route	Displays information about IPv6 static routes.

The following table lists the available **clear** commands for IPv6 networks.

TABLE 10 IPv6 clear commands

Command	Description
clear ipv6 counters	Clears IPv6 counters on all interfaces or on a specified interface.
clear ipv6 neighbor	Clears the IPv6 Neighbor Discovery cache on an interface.
clear ipv6 route	Clears IPv6 routes on an interface.

In addition, you can restrict the flooding of IPv6 multicast data traffic if a multicast group is not learned and instead forward that traffic explicitly to multicast router (mrouter) ports. Those ports are learned either by means of MLD queries or PIMv6 hello-based mrouter detection (both of which are supported by default). This feature is available for multicast profiles only.

To restrict this flooding, use the following command on a VLAN:

ipv6 mld snooping restrict-unknown-multicast

IPv4 Static Routing

Overview of static routing

Static routes are manually configured entries in the IP routing table.

The IP route table can receive routes from several sources, including static routes. Other route sources include directly connected networks, RIP, OSPF, BGP4, and ISIS protocols.

Static routes can be used to specify desired routes, backup routes, or routes of last resort. Static routing can help provide load balancing and can use routing information learned from other protocols.

In setting up static routes, you can specify several types of destinations:

- Destination network, using an IP address and prefix length
- Default network route
- Next-hop router
- Ethernet interface, typically used for directly attached destination networks
- Virtual interface
- Null interface

You can influence the preference a route is given in the following ways:

- By setting a route metric higher than the default metric
- By giving the route an administrative distance
- By specifying a route tag for use with a route map.

Static routes can be configured to serve as any of the following:

- Default routes
- Primary routes
- Backup routes
- Null routes for intentionally dropping traffic when the desired connection fails
- Alternative routes to the same destination to help load balance traffic.

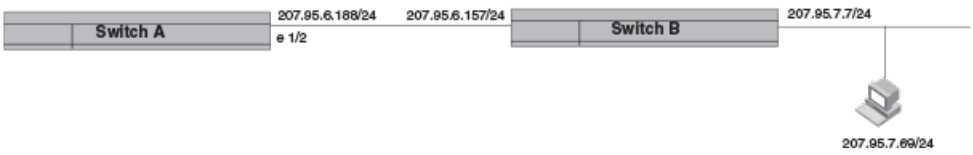
Static route states follow port states

IP static routes remain in the IP route table only as long as the port or virtual interface used by the route is available and the next-hop IP address is valid; otherwise, the software removes the static route from the IP route table. If the port or virtual routing interface becomes available again later and the next-hop is valid, the software adds the route back to the route table.

This feature allows the router to adjust to changes in network topology. The router does not continue trying to use routes on unavailable paths but instead uses routes only when their paths are available.

In the following example, a static route is configured on Switch A. The route configuration is shown following the figure.

FIGURE 5 Example of static route



The following command configures a static route to 207.95.7.0 destinations, using 207.95.6.157 as the next-hop gateway.

```
device(config)# ip route 207.95.7.0/24 207.95.6.157
```

When you configure a static IP route, you specify the destination address for the route and the next-hop gateway or Layer 3 interface through which the Layer 3 device can reach the route. The device adds the route to the IP route table. In this case, Switch A knows that 207.95.6.157 is reachable through port 1/2, and also assumes that local interfaces within that subnet are on the same port. Switch A deduces that IP interface 207.95.7.7 is also on port 1/2.

The software automatically removes a static IP route from the IP route table if the port used by that route becomes unavailable or the IP address is not valid. When the port becomes available again, the software automatically re-adds the route to the IP route table.

Configuring a basic IP static route

To configure a basic IP static route, perform these steps.

1. Enter global configuration mode.

   ```
   device# configure terminal
   ```

2. Enter the Rbridge ID.

   ```
   device(config)# rbridge-id 30
   ```

3. Enter the IP address and prefix length for the route destination network. On the same command line, enter the IP address for the next hop.

```
device(config-rbridge-id-30)# ip route 10.0.0.0/8 10.1.1.1
```

   This example configures an IP static route with a destination network address of 10.0.0.0, a prefix length of /8, and a next hop address of 10.1.1.1.

   NOTE
   Prefix lengths must be used as part of the address. Network masks cannot be used. The prefix length of /8 is equivalent to a network mask of 255.0.0.0. The prefix length of /24 (equivalent to the mask 255.255.255.0) matches all hosts within the Class C subnet address specified in the destination IP address.

The following example configures an IP static route on Rbridge ID 30 with a destination network address of 10.0.0.0, a prefix length of 24 bits, and a next hop address of 10.1.1.1.

```
device# configure terminal
device(config)# rbridge-id 30
device(config-rbridge-id-30)# ip route 10.0.0.0/24 10.1.1.1
```

Adding metrics to a static route

You can influence route preference by adding a cost metric or an administrative distance to a static route.

Follow these steps to create an IP static route with cost metrics.

1. Enter global configuration mode.

```
device# configure terminal
```

2. Enter the Rbridge ID.

```
device(config)# rbridge-id 30
```

3. Designate the route destination and next hop, and add a route priority parameter.

Option	Description
Cost metric	The value is compared to the metric for other static routes in the IPv4 route table to the same destination. Two or more routes to the same destination with the same metric load share traffic to the destination. The value may range from 1 through 16. The default is 1. A route with a cost of 16 is considered unreachable.
Administrative distance	This value is compared to the administrative distance of all routes to the same destination. Static routes by default take precedence over learned protocol routes. However, to give a static route a lower priority than a dynamic route, give the static route the higher administrative distance. The value is preceded by the keyword **distance** and can range from 1 to 254. The default is 1. A value of 255 is considered unreachable.

```
device(config-rbridge-id-30)# ip route 10.128.2.71/24 10.111.10.1 distance 10
```

This example configures a static route with an administrative distance of 10.

NOTE
The device replaces a static route if it receives a route to the same destination with a lower administrative distance.

```
device(config-rbridge-id-30)# ip route 10.128.2.69/24 10.111.10.1 2
```

This example configures a static route with a metric of 2.

The following example configures a static route to destinations with an IP address beginning with 10.0.0.0. The route uses IP address 10.111.10.1 as the next hop. The static route is assigned an administrative distance of 3.

```
device# configure terminal
device(config)# rbridge-id 30
device(config-rbridge-id-30)# ip route 10.0.0.0/24 10.111.10.1 distance 3
```

Configuring a physical interface as next hop

The interface you use for the static route's next hop must have at least one IP address configured on it. The address does not need to be in the same subnet as the destination network.

NOTE
ARP will be generated for a forwarded packet destination IP address when an interface is configured as the next hop.

To configure an IP static route with an IP physical interface as the next hop, follow these steps.

1. Enter global configuration mode.

```
device# configure terminal
```

2. Enter the Rbridge ID.

```
device(config)# rbridge-id 30
```

3. Enter the IP address and prefix-length for the route destination network. On the same command line, enter the appropriate **gigabitethernet** keyword for the line speed followed by the Rbridge ID slot and port of the interface number to be used as next hop.

```
device(config-rbridge-id-30)# ip route 10.128.2.69/24 tengigabitethernet 4/1
```

   This example configures an IP static route with a destination network address of 10.128.2.69 and a prefix-length of 24 on a 10-Gbps connection to the next-hop interface 4/1.

The following example configures an IP static route with a destination network address of 10.128.2.69 and a prefix-length of 24 on a 10-Gbps connection to the next-hop interface 30/4/1.

```
device# configure terminal
device(config)# rbridge-id 30
device(config-rbridge-id-30)# ip route 10.128.2.69/24 tengigabitethernet 4/1
```

Configuring a virtual interface as next hop

The virtual interface you use for the static route's next hop must have at least one IP address configured on it. The address does not need to be in the same subnet as the destination network.

To configure an IP static route that uses a virtual interface as the next hop, follow these steps.

1. Enter global configuration mode.

```
device# configure terminal
```

2. Enter the Rbridge ID.

```
device(config)# rbridge-id 30
```

3. Enter the IP destination address and the prefix-length. On the same command line, enter the keyword **ve** followed by the appropriate VLAN number.

   NOTE
   After the keyword **ve**, you can also enter an interface number (slot/port) or 0 to designate a null route.

```
device(config-rbridge-id-30)# ip route 10.128.2.0/24 ve 3
```

The following example configures an IP static route with a destination address of 10.128.2.0, a prefix-length of /24, and a virtual interface (ve 3) as the next hop.

```
device# configure terminal
device(config)# rbridge-id 30
device(config-rbridge-id-30)# ip route 10.128.2.0/24 ve 3
```

Configuring a static route with a VRF as next hop

A VRF can be designated as the next hop toward a destination via a valid port, IP address, or virtual interface.

The VRF must be an existing VRF, and any physical port referenced in the next hop must have a valid IP address for the route to be added to the routing table.

To configure an IP static route that uses a VRF as the next hop, follow these steps.

1. Enter global configuration mode.

```
device# configure terminal
```

2. Enter the Rbridge ID.

```
device(config)# rbridge-id 30
```

3. Enter the IP destination address and the prefix-length. On the same command line, enter the keyword **next-hop-vrf** followed by the appropriate VRF name. Then specify the next hop as an IP address, physical interface, virtual interface, or null route.

   NOTE
   Using a VRF with a physical port as next hop allows for VRF route leaking for all incoming traffic intended for the destination IP address.

```
device(config-rbridge-id-30)# ip route 10.0.0.0/24 next-hop-vrf red port-channel 1
```

   This example routes traffic for IP address 10.0.0.0/24 through VRF red over port channel 1.

```
device(config-rbridge-id-30)# ip route 10.0.0.0/24 next-hop-vrf red 11.1.2.3
```

   This example routes traffic for IP address 10.0.0.0/24 through VRF red via IP address 11.1.2.3.

```
device(config-rbridge-id-30)# ip route 10.0.0.0/24 next-hop-vrf red ve 3
```

   This example routes traffic for IP address 10.0.0.0/24 through VRF red over virtual interface 3.

The following example creates a static route to destinations with 10.0.0.0 IP addresses through VRF red via 40-Gbps port 30/2/1.

```
device(config-rbridge-id-30)# ip route 10.0.0.0/24 next-hop-vrf red fortygigabitethernet 2/1
```

Configuring a static route for use with a route map

You can configure a static route with a tag that can be referenced in a route map.

Perform these steps to configure a static route with a tag that can be referenced in a route map.

1. Enter global configuration mode.

```
device# configure terminal
```

2. Enter the Rbridge ID.

```
device(config)# rbridge-id 30
```

3. Enter the **ip route** command followed by the destination network IP address and prefix-length and the next-hop IP address. On the same line, enter the keyword **tag** followed by a decimal tag number.

```
device(config-rbridge-id-30)# ip route 10.0.0.0/24 10.1.1.1 tag 3
```

The following example creates an IP static route to destination IP addresses beginning with 10.0.0.0 through the next-hop address 10.1.1.1. The static route includes the tag "3" for later use in a route map.

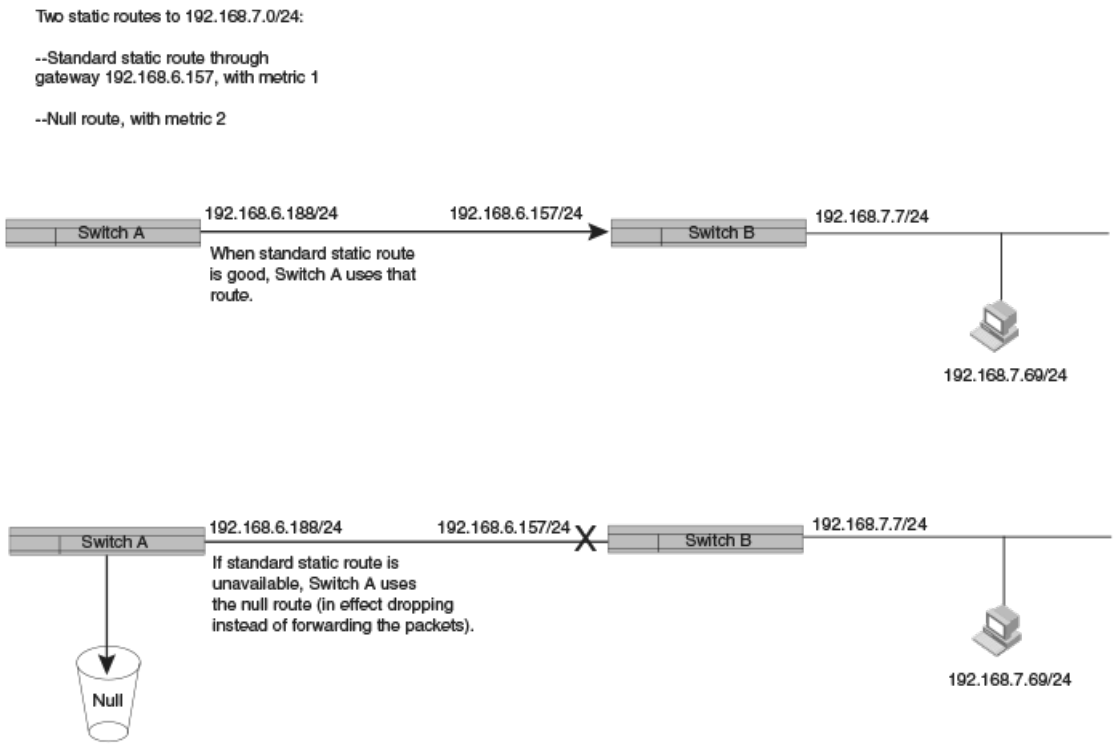
```
device# configure terminal
device(config)# rbridge-id 30
device(config-rbridge-id-30)# ip route 10.0.0.0/24 10.1.1.1 tag 3
```

Configuring a null route

You can configure a null static route to drop packets to a certain destination. This is useful when the traffic should not be forwarded if the preferred route is unavailable.

The following figure depicts how a null static route works with a standard route to the same destination.

FIGURE 6 Null route and standard route to same destination



To configure a null route with a lower priority than the preferred route, perform the following steps.

1. Enter global configuration mode.

   ```
   device# configure terminal
   ```

2. Enter the Rbridge ID.

   ```
   device(config)# rbridge-id 30
   ```

3. Configure the preferred route to a destination.

   ```
   device(config-rbridge-id-30)# ip route 192.168.7.0/24 192.168.6.157
   ```

   This example creates a static route to destination network addresses that have an IP address beginning with 192.168.7.0. These destinations are routed through the next-hop gateway 192.168.6.157. The route carries the default metric of 1.

4. Configure a route to the same destination, followed by the keyword **null 0**. On the same line, enter a cost metric that is higher than the metric for the preferred route.

```
device(config-rbridge-id-30)# ip route 192.168.7.0/24 null 0 2
```

   This example creates a null static route to the same destination. The metric is set to 2, which is higher than the metric for the preferred route. When the preferred route becomes unavailable, the null route is used, and traffic to the destination is dropped.

The following example creates a primary route to all destinations beginning with 192.168.7.0. It creates an alternative null route to drop the packets when the primary route is not available.

```
device# configure terminal
device(config)# rbridge-id 30
device(config-rbridge-id-30)# ip route 192.168.7.0/24 192.168.6.157
device(config-rbridge-id-30)# ip route 192.168.7.0/24 null 0 2
```

Configuring a default static route

You can manually create a default static route that the router uses if there are no other default routes to a destination.

Perform these steps to configure a default route.

1. Enter global configuration mode.

```
device# configure terminal
```

2. Enter the Rbridge ID.

```
device(config)# rbridge-id 30
```

3. Enter 0.0.0.0/0 as the destination route and prefix-length. On the same line, enter a valid next-hop address.

```
device(config-rbridge-id-30)#  ip route 0.0.0.0/0 10.24.4.1
```

   The example creates a default route through next-hop address 10.24.4.1.

The following example configures the network default static route through next-hop 10.24.4.1.

```
device# configure terminal
device(config)#
device(config)# rbridge-id 30
device(config-rbridge-id-30)# ip route 0.0.0.0/0 10.24.4.1
```

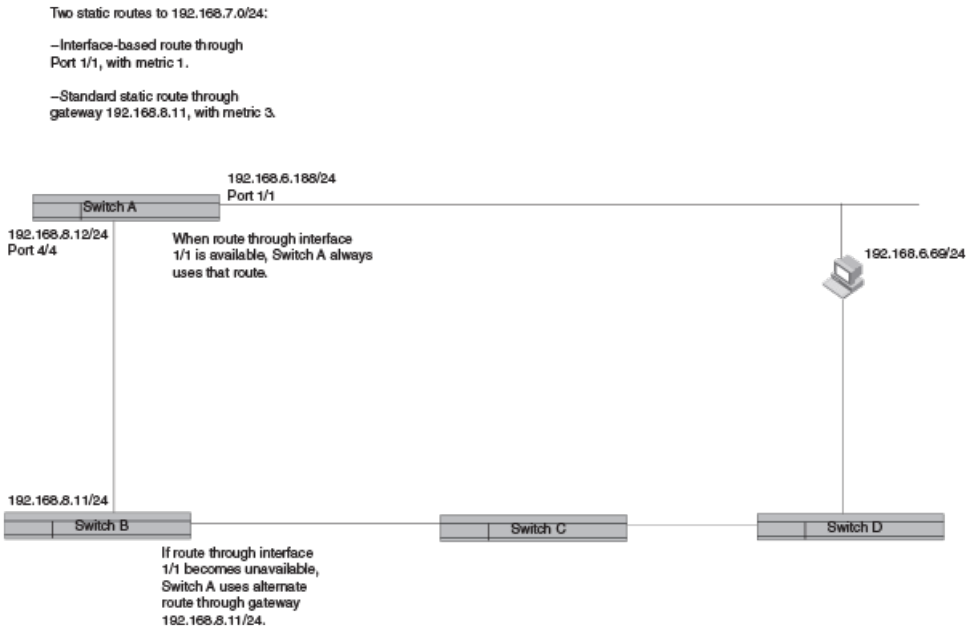
Configuring load sharing and redundancy

You can configure multiple IP static routes to the same destination to set up load sharing or backup routes.

If you configure more than one static route to the same destination with different next-hop gateways but the same metrics, the router load balances among the routes using a basic round-robin method.

If you configure multiple static IP routes to the same destination with different next-hop gateways and different metrics, the router always uses the route with the lowest metric. If this route becomes unavailable, the router fails over to the static route with the next-lowest metric. The following figure depicts two routes with different metrics configured for the same destination.

FIGURE 7 Two static routes to same destination



To set up multiple routes for load sharing or redundancy, perform the following steps.

NOTE

You can also use administrative distance to set route priority; however, be sure to give a static route a lower administrative distance than other types of routes, unless you want other route types to be preferred over the static route.

1. Enter global configuration mode.

```
device# configure terminal
```

2. Enter the Rbridge ID.

```
device(config)# rbridge-id 30
```

3. Enter multiple routes to the same destination using different next hops.

```
device(config-rbridge-id-30)# ip route 10.128.2.0/24 10.157.22.1
device(config-rbridge-id-30)# ip route 10.128.2.0/24 10.111.10.1
device(config-rbridge-id-30)# ip route 10.128.2.0/24 10.1.1.1
```

This example creates three next-hop gateways to the destination. Traffic will alternate among the three paths through next-hop 10.157.22.1, next-hop 10.111.10.1, and next hop 10.1.1.1.

4. To prioritize the three routes, use different metrics for each of the three potential next hops.

```
device(config-rbridge-id-30)# ip route 10.128.2.0/24 10.157.22.1
device(config-rbridge-id-30)# ip route 10.128.2.0/24 10.111.10.1 2
device(config-rbridge-id-30)# ip route 10.128.2.0/24 10.1.1.1 3
```

This example creates three alternate routes to the destination. The primary next hop is 10.157.22.1, which has the default metric of 1 (the default metric is not entered in the CLI). If this path is not available, traffic is directed to 10.111.10.1, which has the next lowest metric of 2. If the second path fails, traffic is directed to 10.1.1.1, which has a metric of 3.

Displaying IPv4 static routes

You can check configured IPv4 routes, static routes, directly connected routes, routes configured for different protocols, the cost associated with each route, and the time the route has been available.

1. To display a list of active static routes and their connection times, at the device prompt, enter the **show ip route static** command.
2. To show all active IP routes and their connection times, enter the **show ip route** command.

The following example shows seven IP routes. Three routes are direct routes, and two of those are connected. All other routes are reached via gateway 1.1.1.2 or 1.1.2.2.

```
device# show ip route
 Total number of IP routes: 7
Type Codes - B:BGP D:Connected O:OSPF R:RIP S:Static U: Unnumbered +: Leaked route; Cost - Dist/Metric
BGP  Codes - i:iBGP e:eBGP
OSPF Codes - i:Inter Area 1:External Type 1 2:External Type 2 s:Sham Link
        Destination        Gateway         Port        Cost          Type Uptime
1       1.1.1.0/24         DIRECT          Te 2/1      0/0           D    1m57s
2       1.1.2.0/24         DIRECT          Te 2/2      0/0           D    0m6s
3       100.1.1.0/24       1.1.1.2         Te 2/1      1/1           S    1m32s
4       100.1.2.0/24       1.1.1.2         Te 2/1      1/1           S    1m16s
5       100.1.3.0/24       1.1.1.2         Te 2/1      1/1           S    1m13s
6       100.2.1.0/24       DIRECT          Te 2/1      1/1           S    0m57s
7       100.3.1.0/24       1.1.1.2         Te 2/1      1/1           S    0m5s
        100.3.1.0/24       1.1.2.2         Te 2/2      1/1           S    0m5s
```

IPv6 Static Routing

Overview of static routing

Static routes can be used to specify desired routes, backup routes, or routes of last resort. Static routing can help provide load balancing.

Static routes are manually configured entries in the existing IPv6 routing table. In setting up static routes, you can specify several types of destinations:

- Destination network, using an IP address and prefix length
- Default network route
- VRF for the next hop
- Ethernet interface, typically used for directly attached destination networks
- Virtual interface
- Null interface

You can influence the preference a route is given in the following ways:

- By setting a route metric higher than the default metric
- By giving the route an administrative distance

Static routes can be configured to serve as any of the following:

- Default routes
- Primary routes
- Backup routes
- Null routes for intentionally dropping traffic when the desired connection fails
- Alternative routes to the same destination to help load balance traffic.

Static route states follow port states

IP static routes remain in the IP route table only as long as the port or virtual interface used by the route is available. If the port or virtual routing interface becomes unavailable, the software removes the static route from the IP route table. If the port or virtual routing interface becomes available again later, the software adds the route back to the route table. This feature allows the router to adjust to changes in network topology. The router does not continue trying to use routes on unavailable paths but instead uses routes only when their paths are available.

In the following example, a static route is configured on Switch A.

FIGURE 8 Example of static route

The following command configures a static route to 2001:DB8::0/32, using 2001:DB8:2343:0:ee44::1 as the next-hop gateway.

```
device(config)# ipv6 route 2001:DB8::0/32 2001:DB8:2343:0:ee44::1
```

When you configure a static IP route, you specify the destination address for the route and the next-hop gateway or Layer 3 interface through which the Layer 3 device can reach the route. The device adds the route to the IP route table. In this case, Switch A knows that 2001:DB8:2343:0:ee44::1 is reachable through port 1/2, and also assumes that local interfaces within that subnet are on the same port. Switch A deduces that IP interface 2001:DB8::0/32 is also on port 1/2.

Configuring a basic IPv6 static route

To configure a basic IPv6 static route, specify the IPv6 destination address, the address mask, and the IPv6 address of the next hop.

To configure a basic IPv6 static route, perform these steps.

1. Enter global configuration mode.

   ```
   device# configure terminal
   ```

2. Enter the Rbridge ID.

   ```
   device(config)# rbridge-id 30
   ```

3. Designate the route destination as an IPv6 address in hexadecimal with 16-bit values between colons, as specified in RFC 2373, and include the address prefix length preceded by a slash. On the same command line, enter the IPv6 address of the next-hop gateway.

   ```
   device(config-rbridge-id-30)# ipv6 route 2001:DB8::0/32 2001:DB8:0:ee44::1
   ```

The following example configures an IPv6 static route for a destination network with the prefix 2001:DB8::0/32 and a next-hop gateway with the global address 2001:DB8:0:ee44::1

```
device# configure terminal
device(config)# rbridge-id 30
device(config-rbridge-id-30)# ipv6 route 2001:DB8::0/32 2001:DB8:0:ee44::1
```

Removing an IPv6 static route

The **no** form of the ipv6 route command must be entered with exact parameters to remove the command. If the route is configured in a non-default VRF, the **no** form of the ipv6 route command must be entered in VRF configuration mode.

Follow these steps to remove an IPv6 static route.

1. (Optional) To view configured routes and confirm exact parameters, enter the command **show ipv6 route** to display the IPv6 route table.

```
device# show ipv6 route
```

2. (Optional) Enter the **show ipv6 static route** command to narrow the output to static routes only.

```
device# show ipv6 static route
```

3. Enter global configuration mode.

```
device# configure terminal
```

4. Enter the Rbridge ID.

```
device(config)# rbridge-id 30
```

5. Enter **no** followed by the **ipv6 route** command, including destination and next-hop, as shown in the following example. (You do not need to include cost metric, distance, or tag parameters.)

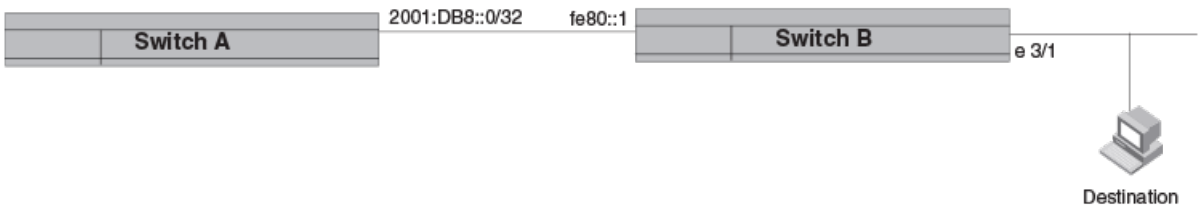
```
device(config-rbridge-id-30)# no ipv6 route 2224::1/128 fe80::205:33ff:fee6:a501 ve 2
```

This example removes the IPv6 route specified.

Configuring an interface as next hop

To configure an IPv6 static route with an interface as the next hop as depicted in the following illustration, perform these steps.

FIGURE 9 IPv6 static route with an interface as next hop



1. Enter global configuration mode.

```
device# configure terminal
```

2. Enter the Rbridge ID.

```
device(config)# rbridge-id 30
```

3. Designate the route destination, followed by the next-hop link-local IPv6 address. On the same command line, enter the correct keyword for the line speed (**gigabitethernet**, **tengigabitethernet**, **fortygigabitethernet**, or **hundredgigabitethernet**) followed by the interface (port/slot) to serve as the next-hop.

```
device(config-rbridge-id-30)# ipv6 route 2001:DB8::0/32 fe80::1 tengigabitethernet 3/1
```

The following example configures a static IPv6 route for a destination network with the prefix 2001:DB8::0/32 and a next-hop gateway with the link-local address fe80::1 that the Layer 3 switch can access through 10G interface 3/1.

```
device# configure terminal
device(config)# rbridge-id 30
device(config-rbridge-id-30)# ipv6 route 2001:DB8::0/32 fe80::1 tengigabitethernet 3/1
```

Configuring a virtual interface as next hop

To configure a basic IPv6 static route with a virtual interface as a next hop, perform these steps.

1. Enter global configuration mode.

   ```
   device# configure terminal
   ```

2. Enter the Rbridge ID.

   ```
   device(config)# rbridge-id 30
   ```

3. Enter the IP address prefix and prefix length for the route destination network.

   ```
   device(config-rbridge-id-30)# ipv6 route 2001:DB8::0/32
   ```

   This example shows the first half of the command, the route destination, IPv6 2001:DB8::0/32 network addresses.

4. On the same command line, add the keyword **ve** followed by the virtual interface ID to be used as the next hop.

   ```
   device(config-rbridge-id-30)# ipv6 route 2001:DB8::0/32 ve 3
   ```

The following example configures an IPv6 static route to IPv6 2001:DB8::0/32 destinations through next-hop virtual interface 3.

```
device# configure terminal
device(config)# rbridge-id 30
device(config-rbridge-id-30)# ipv6 route 2001:DB8::0/32 ve 3
```

Configuring a VRF as next hop for an IPv6 static route

A non-default VRF can be configured as the next-hop gateway for an IPv6 static route.

NOTE
The VRF designated in the procedure must be a valid VRF.

To configure a VRF as the next hop for an IPv6 static route, follow these steps.

1. Enter global configuration mode.

   ```
   device# configure terminal
   ```

2. Enter the Rbridge ID.

   ```
   device(config)# rbridge-id 30
   ```

3. Enter the **ipv6 route** command followed by the IPv6 destination address and prefix length. On the same command line, enter the keyword **next-hop-vrf** followed by the name of the VRF that contains the next-hop gateway router and its IPv6 address.

```
device(config-rbridge-id-30)# ipv6 route 2001:DB8::0/32 next-hop-vrf partners 2001:DB8:0:ee44::1
```

This example creates an IPv6 static route to IPv6 2001:DB8::0/32 destinations through the VRF named "partners" and the next-hop router with the IPv6 address 2001:DB8:0:ee44::1.

Adding metrics to an IPv6 static route

You can influence how likely a static route is to be used by modifying the cost metric or the administrative distance.

Follow these steps to create an IPv6 static route with cost metrics.

1. Enter global configuration mode.

```
device# configure terminal
```

2. Enter the Rbridge ID.

```
device(config)# rbridge-id 30
```

3. Designate the route destination and next hop, and add a route priority parameter.

Option	Description
Cost metric	The value is compared to the metric for other static routes in the IPv6 route table to the same destination. Two or more routes to the same destination with the same metric load share traffic to the destination. The value may range from 1 through 16. The default is 1. A route with a cost of 16 is considered unreachable.
Administrative distance	This value is compared to the administrative distance of all routes to the same destination. Static routes by default take precedence over learned protocol routes. However, to give a static route a lower priority than a dynamic route, give the static route the higher administrative distance. The value is preceded by the keyword **distance** and can range from 1 to 254. The default is 1. A value of 255 is considered unreachable.

```
device(config-rbridge-id-30)# ipv6 route 2001:DB8::0/64 2001:DB8:2343:0:ee44::1 2
```

This example configures a static IPv6 route for a destination network with the prefix 2001:DB8::0/64 and a next-hop gateway with the global address 2001:DB8:0:ee44::1 and assigns the route a metric of 2.

```
device(config-rbridge-id-30)# ipv6 route 2001:DB8::0/64 2001:DB8:2343:0:ee44::1 distance 254
```

This example configures a static route with an administrative distance of 254.

The following example configures an IPv6 static route for a destination network with the prefix 2001:DB8::0/64 and a next-hop gateway with the global address 2001:DB8:0:ee44::1. The static route is assigned an administrative distance of 3.

```
device# configure terminal
device(config-rbridge-id-30)# ipv6 route 2001:DB8::0/64 2001:DB8:0:ee44::1 distance 3
```

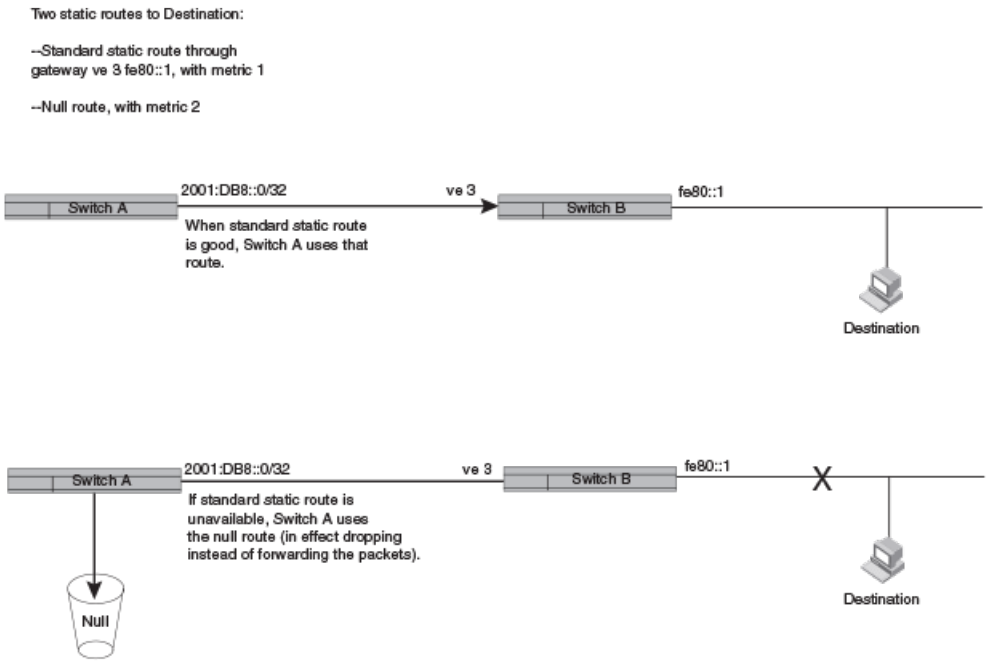
Configuring a null route

You can configure a null static route to drop packets to a certain destination. This is useful when the traffic should not be forwarded if the preferred route is unavailable.

> NOTE
> You cannot add a null or interface-based static route to a network if there is already a static route of any type with the same metric you specify for the null or interface-based route.

The following figure depicts how a null static route works with a standard route to the same destination.

FIGURE 10 Null route and standard route to same destination



The following procedure creates a preferred route and a null route to the same destination. The null route drops packets when the preferred route is not available.

1. Enter global configuration mode.

   ```
   device# configure terminal
   ```

2. Enter the Rbridge ID.

   ```
   device(config)# rbridge-id 30
   ```

3. Configure the preferred route to a destination.

   ```
   device(config-rbridge-id-30)# ipv6 route 2001:DB8::0/64 ve 3
   ```

   This example creates a static route to IPv6 2001 : DB8 : : 0/64 destination addresses. These destinations are routed through link-local address fe80::1 and the next hop gateway virtual interface (ve) 3. The route uses the default cost metric of 1.

4. Configure the null route to the same destination with a higher metric.

```
device(config-rbridge-id-30)# ipv6 route 2001:DB8::0/64 null 0 2
```

   This example creates a null static route to the same destination. The metric is set higher so that the preferred route is used if it is available.

The following example creates a primary route to all 2001 : DB8 : : 0/64 destinations through virtual interface (ve) 3. It creates an alternative null route to drop the packets when the primary route is not available.

```
device# configure terminal
device(config)# rbridge-id 30
device(config-rbridge-id-30)# ipv6 route 2001 : DB8 : : 0/64 ve 3
device(config-rbridge-id-30)# ipv6 route 2001 : DB8 : : 0/64 null 0 2
```

Configuring a default static route

You can manually create a default static route that the router uses if there are no other default routes to a destination.

Perform these steps to configure a default route.

1. Enter global configuration mode.

```
device# configure terminal
```

2. Enter the Rbridge ID.

```
device(config)# rbridge-id 30
```

3. Enter the following destination route and network mask followed by a valid next-hop address.

```
device(config-rbridge-id-30)#  ipv6 route ::/0 2001:DB8:0:ee44::1
```

The following example configures a default static route to global IPv6 address 2001:DB8:0:ee44::1.

```
device# configure terminal
device(config)# rbridge-id 30
device(config-rbridge-id-30)# ipv6 route ::/0 2001:DB8:0:ee44::1
```

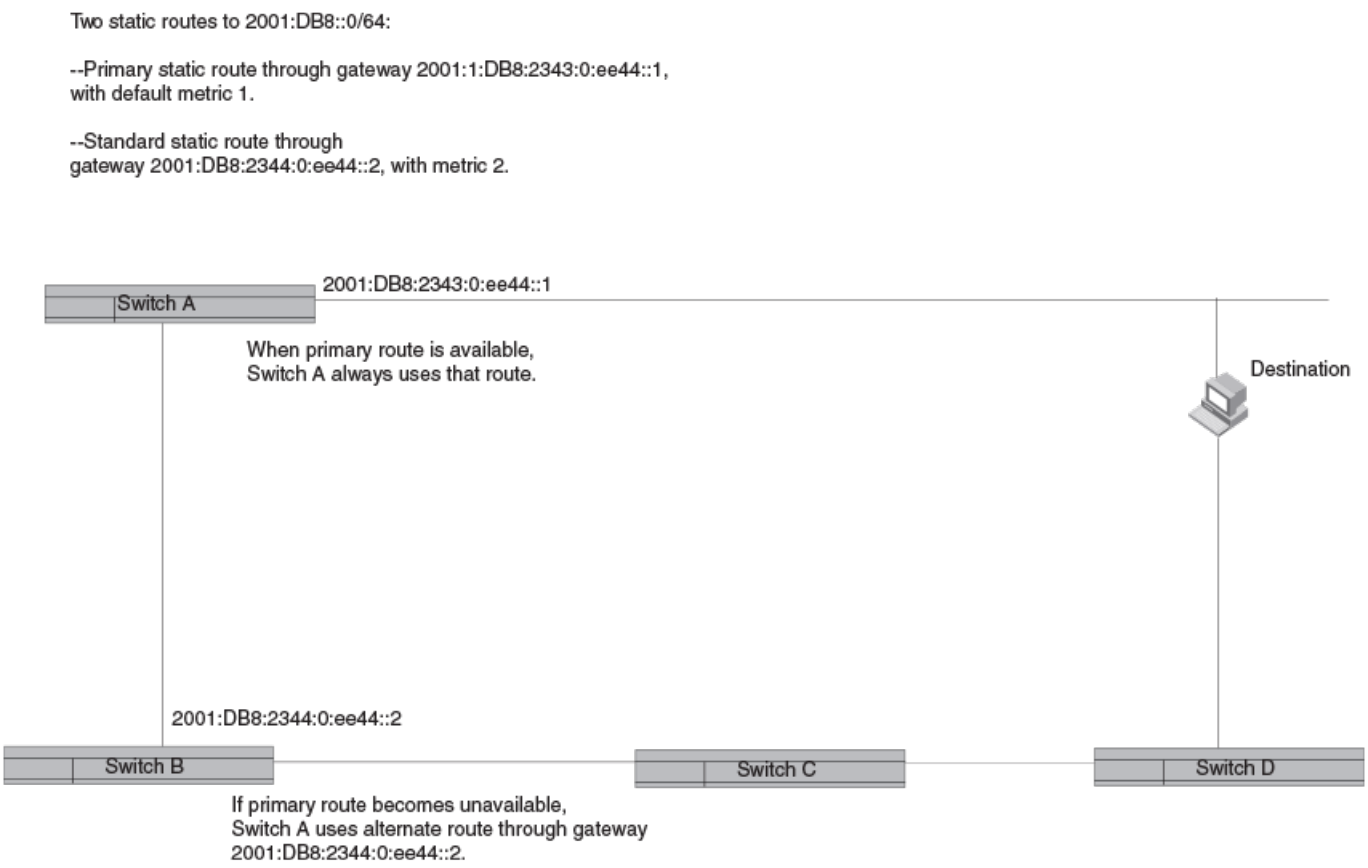
Configuring load sharing and redundancy

You can configure multiple IP static routes to the same destination to set up load sharing or backup routes.

If you configure more than one static route to the same destination with different next-hop gateways but the same metrics, the router load balances among the routes using a basic round-robin method.

If you configure multiple static IP routes to the same destination with different next-hop gateways and different metrics, the router always uses the route with the lowest metric. If this route becomes unavailable, the router fails over to the static route with the next-lowest metric. The following figure depicts multiple routes with different metrics configured for the same destination.

FIGURE 11 Two static routes to same destination



To set up multiple routes for load sharing or redundancy, perform the following steps.

NOTE
You can also use administrative distance to set route priority; however, be sure to give a static route a lower administrative distance than other types of routes, unless you want other route types to be preferred over the static route.

1. Enter global configuration mode.

```
device# configure terminal
```

2. Enter the Rbridge ID.

```
device(config)# rbridge-id 30
```

3. Enter multiple routes to the same destination using different next hops.

```
device(config-rbridge-id-30)# ipv6 route 2001:DB8::0/64 2001:DB8:2343:0:ee44::1
device(config-rbridge-id-30)# ipv6 route 2001:DB8::0/64 2001:DB8:2344:0:ee44::2
```

This example creates two next-hop gateways for all 2001:DB8::0/64 destinations. Traffic will alternate between the two paths.

4. To prioritize multiple routes, use different metrics for each possible next hop.

```
device(config-rbridge-id-30)# ipv6 route 2001:DB8::0/64 2001:DB8:2343:0:ee44::1
device(config-rbridge-id-30)# ipv6 route 2001:DB8::0/64 2001:DB8:2344:0:ee44::2 2
```

   This example creates an alternate route to all 2001:DB8::0/64 destinations. The primary route uses 2001:DB8:2343:0:ee44::1 as the next hop. The route has the default metric of 1. If this path is not available, traffic is directed through 2001:DB8:2344:0:ee44::2, which has the next lowest metric (2).

Adding an IPv6 static route tag for use with route-maps

To configure an IPv6 static route with a tag that can be referenced in a route-map, follow these steps.

1. Enter global configuration mode.

```
device# configure terminal
```

2. Enter the Rbridge ID.

```
device(config)# rbridge-id 30
```

3. Configure the IPv6 static route destination address and next-hop address. On the same command line, enter the keyword tag, followed by the decimal number to be referenced later in a route-map.

```
device(config-rbridge-id-30)# ipv6 route 2001:DB8::0/64 2001:DB8:0:ee44::1 tag 3
```

The following example configures an IPv6 route to IPv6 2001:DB8::0/64 destinations through next-hop 2001:DB8:0:ee44::1. The route has the tag ID "3," which can be referenced later in a route-map.

```
device# configure terminal
device(config)# rbridge-id 30
device(config-rbridge-id-30)# ipv6 route 2001:DB8::0/64 2001:DB8:0:ee44::1 tag 3
```

DHCPv4

DHCPv4 overview

The Dynamic Host Configuration Protocol for DHCPv4 enables DHCP servers to pass configuration parameters such as IPv4 addresses to IPv4 hosts.

The Dynamic Host Configuration Protocol (DHCP) is based on the Bootstrap Protocol (BOOTP) and provides configuration parameters such as IP addresses, default routes, DNS server addresses, access control, QoS policies, and security policies stored in DHCP server databases to DHCP clients upon request. DHCP enables the automatic configuration of client systems. DHCP removes the need to configure devices individually. Clients can set network properties by connecting to the DHCP server instead. This protocol consists of two components; a protocol to deliver host-specific configuration parameters from a DHCP server to a host, and a mechanism to allocate network addresses to hosts.

DHCP is built on a client-server model, where designated DHCP server hosts allocate network addresses and deliver configuration parameters to dynamically configured hosts.

You can enable DHCP service on VDX switches so that they can automatically obtain an Ethernet IP address, prefix length, and default gateway address from the DHCP server. Refer to the "Configuring an IPv4 address with DHCP" section of the *Extreme Network OS Management Configuration Guide* for more information.

DHCP protocol

Dynamic Host Configuration Protocol (DHCP) is an IP network protocol that provides network configuration data, such as IP addresses, default routes, DNS server addresses, access control, QoS policies, and security policies stored in DHCP server databases to DHCP clients upon request.

You can enable DHCP service on VDX switches so that they can automatically obtain an Ethernet IP address, prefix length, and default gateway address from the DHCP server. Refer to the "Configuring an IPv4 address with DHCP" section of the *Extreme Network OS Management Configuration Guide* for more information.

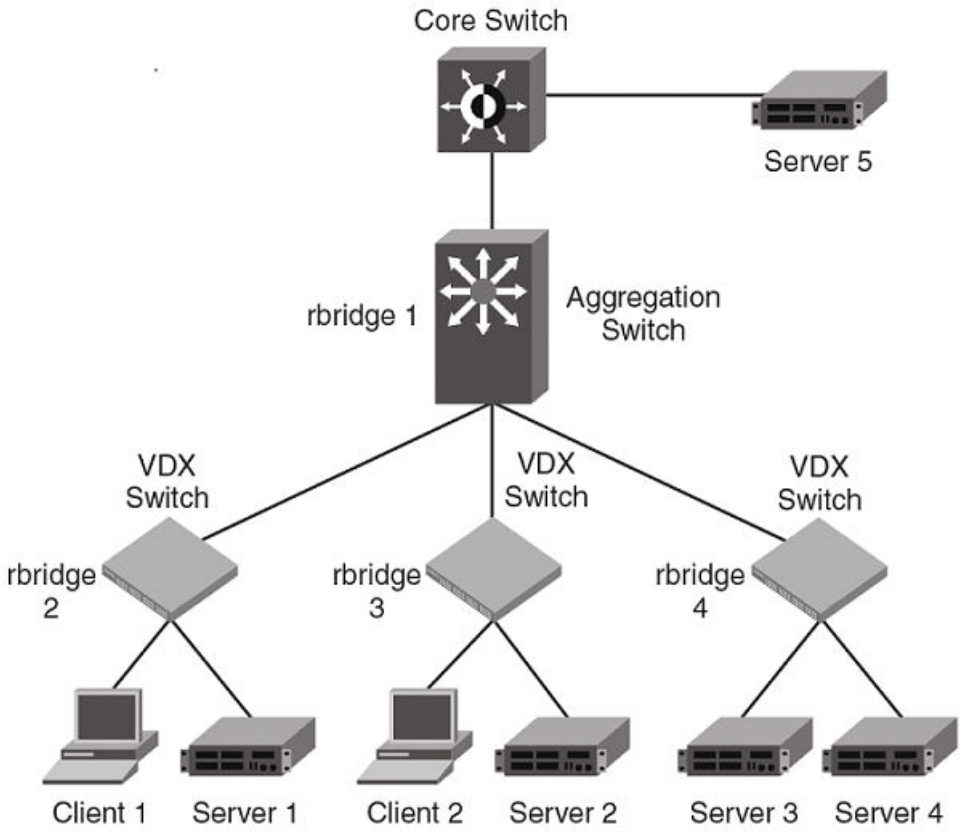
IP DHCP relay overview

The IP DHCP relay feature allows forwarding of requests and replies between DHCP servers and clients connected to the switch when these servers and clients are not on the same subnet.

You can configure the IP DHCP relay feature on any Layer 3 interface to forward requests and replies between DHCP servers and clients connected to the switch when these servers and clients are not on the same subnet.

A Layer 3 interface could be the switch front-end Ethernet interface (VE port), port channel, or physical interface.

The following figure shows an example of a VCS cluster configuration with DHCP servers and clients located on different subnets.



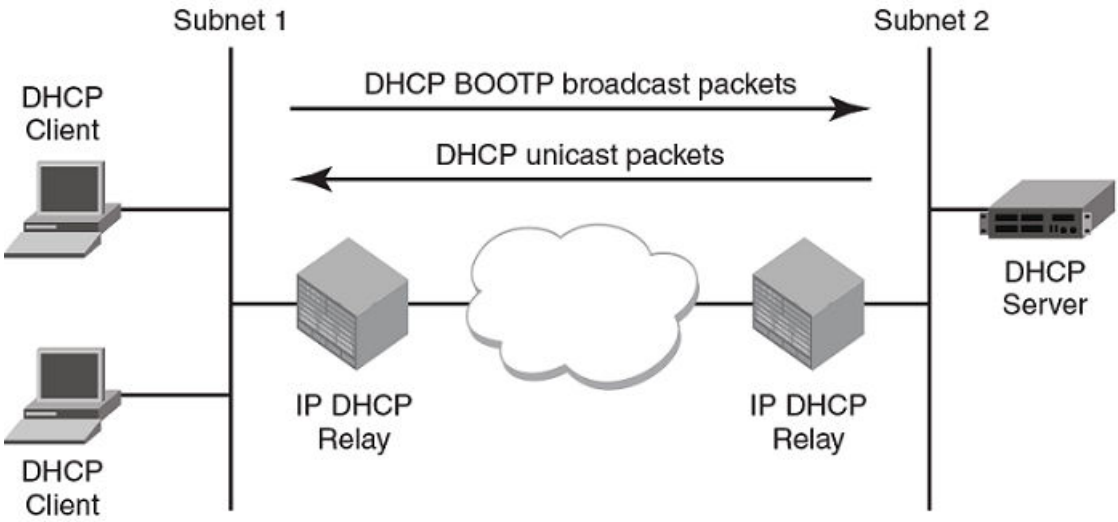
The previous figure illustrates a VCS cluster with clients and servers passing data between different subnets. Note the following examples of configurations supported and not supported by the IP DHCP Relay feature.

Following are the supported configuration examples:

- Local DHCP server. DHCP Client 1 and Server 1 are on the same RBridge, but on different subnets. This configuration supports the IP DHCP Relay feature.
- Remote DHCP server. Client 1 and Server 2 are on different rbridges, but on different subnets. This configuration supports the IP DHCP Relay feature.
- DHCP server across a WAN. Client 1 and Server 5 are on different subnets across the WAN.
- DHCP server is in a different Virtual Routing and Forwarding (VRF) instance. Client 1 and Server 2 are in different VRFs.

The only unsupported configuration is a Network DHCP server. Client 1 is on a different subnet than Server 3 and Server 4, which are on the same subnet.

The DHCP Relay agent forwards DHCP BOOTP broadcast packets from the DHCP clients to the appropriate server and processes broadcast or unicast packets from the server to forward to the DHCP client. BOOTP is a network protocol used to obtain an IP address from a DHCP server. Refer to the following figure.



Configuring IP DHCP Relay

Configure the IP DHCP Relay agent on any Layer 3 interface using the IP address of the DHCP server where client requests are to be forwarded.

Layer 3 interfaces can be a virtual Ethernet (VE), port channel (PO), or a physical 1, 10, or 40 Gigabit Ethernet interface.

You can configure the IP DHCP Relay agent using the **ip dhcp relay address** command followed by the IP address of the DHCP server. Use the **use-vrf** *vrf-name* parameter if the DHCP server and client interface are on different Virtual Forwarding and Routing (VRF) instances.

The following are considerations and limitations when configuring the IP DHCP Relay agent:

- You can configure up to four DHCP server IP addresses per interface. When multiple addresses are configured, the relay agent relays the packets to all server addresses.
- The DHCP server and clients it communicates with can be attached to different VRF instances. When clients and the DHCP server are on different VRFs, use the **use-vrf** *vrf-name* option with the **ip dhcp relay address** command, where *vrf-name* is the VRF where the DHCP server is located. For more information on VRF support for the IP DHCP Relay, refer to VRF support on page 95.

Perform the following steps to configure an IP DHCP Relay:

1. In privileged EXEC mode, enter the **configure terminal** command to enter the global configuration mode.

```
device# configure
```

2. Enter the rbridge-id command with an RBridge ID to enter the RBridge ID configuration mode.

```
device(config)# rbridge-id 1
```

3. Enter the **interface** command followed by the interface ID to enter the interface configuration mode for the Ve, PO or physical interface where you want to configure the IP DHCP Relay.

```
device(config-rbridge-id-1)# int Ve 101
```

4. Enter the **ip dhcp relay address** *ip-addr* **use-vrf** *vrf-name* command where *ip-addr* is the IP address of the DHCP server. Use the **use-vrf** *vrf-name* option if the DHCP server is on a different VRF instance than the interface where the client is connected.

```
device(config)# ip dhcp relay address 100.1.1.2
```

5. To remove the IP DHCP Relay address enter the **no ip dhcp relay address** *ip-addr* **use-vrf** *vrf-name* command.

The following is an example of configuring two IP DHCP Relay addresses on a physical 1 GbE interface in slot 2, port 4 on RBridge ID 1.

NOTE
In this example, the local DHCP server IP address is 10.1.1.2.

```
device# config
Entering configuration mode terminal

device(config)# rbridge-id 1
device(config-rbridge-id-1)# int Ve 101
device(config-Ve-101)# ip dhcp relay address 100.1.1.2
device(config-Ve-101)# ip dhcp relay address 12.3.4.6
device(config-Ve-101)# ip dhcp relay address 10.1.1.2
```

The following is an example of configuring two IP DHCP Relay addresses on virtual Ethernet interface 102.

NOTE
In this example, the IP address 3.1.1.255 is the local subnet broadcast address. The relay agent relays the DHCP packets to the directed broadcast address and all addresses for DHCP servers configured on the interface.

```
device# config
Entering configuration mode terminal

device(config)# rbridge-id 2
device(config-rbridge-id-2)# int Ve 102
device(config-Ve-102))# ip dhcp relay address 200.1.1.2
device(config-Ve-102)# ip dhcp relay address 10.1.1.255
```

The following example removes an IP DHCP Relay address using the **no** option in the **ip dhcp relay address** *ip-addr* command:

```
device(config-if)# no ip dhcp relay address 200.1.1.2
```

If the DHCP server is on a different VRF than the interface where the client is connected, use the **use-vrf** *vrf-name* option in the **ip dhcp relay address** *ip-addr* command.

NOTE
If the **use-vrf** *vrf-name* option is not used, it is assumed that the DHCP server and client interface are on the same VRF.

```
device# config
Entering configuration mode terminal

device(config)# rbridge-id 2
device(config-rbridge-id-2)# int Ve 103
device(config-Ve-103)# ip dhcp relay address 10.1.2.255 use-vrf blue
```

The following example removes an IP DHCP Relay address using the **no** option in the **ip dhcp relay address** *ip-addr* **use-vrf** *vrf-name* command:

```
device(config-ve-103)# no ip dhcp relay address 10.1.2.255 use-vrf blue
```

DHCP relay agent information option 82

When DHCP relay agent information option 82 is enabled, relay agent information option-82 is inserted by the relay agent before relaying DHCP client requests to the server.

The relay agent information option-82 allows the DHCP server to select a sub-range in the DHCP server address pool. The DHCP server echos the option 82 in the DHCP reply packet. The DHCP relay agent validates and removes the option 82 information, and then sends the response to the DHCP client.

Adding option 82 to the DHCP client helps address the following security issues:

- Allows the relay agent to identify the circuit to which to forward replies.
- Prevents DHCP IP address exhaustion attacks. IP address exhaustion occurs when an attacker requests all available IP addresses from a DHCP server by sending requests with fake client MAC addresses.
- Prevents permanently assigning an IP address to a particular user or modem.
- Prevents spoofing of client identifier fields used to assign IP addresses.
- Prevents denial of service (DoS) attacks.

The DHCP Relay Agent Information Option is a container option for specific agent-supplied sub-options. The format of the relay agent information option is as follows:

Code	Len	Agent Information Field					
82	N	i1	i2	i3	i4	...	iN

NOTE
The length N represents the total number of octets in the Agent Information Field. The Agent Information field consists of a sequence of SubOpt/Length/Value tuples for each sub-option.

Relay Agent Circuit ID sub-option

Sub-option type (1 byte)	Length (1 byte)	VLAN ID <string> (4 bytes)	IF-description string (4 bytes)
2	68		

NOTE
The circuit ID is a combination of the VLAN-ID and the interface description string. If the interface description is not configured, the default string “Brocade” is used in the circuit ID.

Agent remote ID sub-option

Sub-option type (1 byte)	Length (1 byte)	VLAN ID (2 bytes)	MAC address (6 bytes)
2	8		

Relay agent operation with Option 82 enabled

When Option 82 is enabled, the relay agent performs the following actions:

- If the client receives a DHCP packet with the GIADDR field set to zero, but with the Option 82 already present, the relay agent discards the packet and increments the error count.
- If the client receives a DHCP packet with the GIADDR field set to a GIADDR implemented by the local agent, the packet is discarded.
- Adds the IP address of the relay agent (in the GIADDR field).
- Inserts the Option 82 information as the last option in a request packet. Option 82 information contains the remote ID sub-option and the circuit ID sub-option.
- Relays the packet to the DHCP server.
- Removes Option 82 from the received packets from the DHCP server after validation.
- Forwards the packet to the client.

Configuration considerations

The following configuration considerations apply to DHCP Option 82:

- The relay agent does not monitor the client requests during the renewal phase. Also, the device forwards request packets with a non-zero GIADDR from a different relay agent.
- All client interfaces are treated as untrusted interfaces.
- You cannot configure each sub-option separately. Enabling Option 82 enables the insertion of the circuit ID and remote ID sub-options.
- DHCP relay Option 82 can be enabled or disabled globally. You cannot enable or disable this option at the interface level.

Enabling the DHCP Relay Agent Information option

Complete the following steps to enable insertion or removal of DHCP relay information option-82 present in the DHCP client and server packets respectively.

1. Enter the global configuration mode.

```
device# configure
```

2. Enter the rbridge ID configuration mode.

```
device(config)# rbridge-id 1
```

3. Enter the **ip dhcp relay information option** command to enable option-82.

```
device(config-rbridge-id-1)# ip dhcp relay information option
```

4. Specify the interface description (string) to be used for the IPv4 DHCP relay Option 82 Circuit ID for a Ve interface.

```
device(config-Vlan-1)# description dcip
```

5. You can also specify the interface description (string) to be used for the IPv4 DHCP relay Option 82 Circuit ID for a physical router port.

```
device(conf-if-te-201/0/7)# description dcip
```

VRF support

Virtual Routing and Forwarding (VRF) is a technology that controls information flow within a network by partitioning the network into different logical VRF domains to isolate traffic. This allows a single device to have multiple containers of routing tables or Forwarding Information Bases (FIBs), with one routing table for each VRF instance. This permits a VRF-capable router to function as a group of multiple virtual routers on the same physical router.

Inter-VRF route leaking allows leaking of specific route prefixes from one VRF instance to another on the same physical router, which eliminates the need for external routing.

In a DHCP setting, route leaking is controlled through a single DHCP server (which may be on a different VRF). This permits multiple VRFs to communicate with that server.

The IP DHCP Relay is supported in configurations where the DHCP server is on the same or different VRF instances than the interface through which the client is connected. When the DHCP server and client are on different VRFs, this is called inter-VRF deployment. For inter-VRF deployment, use the use-vrf *vrf-name* option with the **ip dhcp relay address** command, where *vrf-name* is the VRF where the DHCP server is located.

For more information on VRFs, refer to the Virtual Routing and Forwarding chapter.

Supported VRF configuration examples

Following are examples of VRF configurations that are supported for IP DHCP Relay:

- Client interface and DHCP server are on the same VRF instance. For example:
  - VE interface 100 in VRF "red"
  - IP address of interface - 3.1.1.1/24
  - IP DHCP Relay address (20.1.1.2) in VRF "red"
- Client interface and DHCP servers are on different VRF instances. For example:
  - VE interface 100 in default VRF
  - IP address of interface - 3.1.1.1/24
  - IP DHCP Relay address (100.1.1.2) in VRF "blue"
  - IP DHCP Relay address (1.2.3.4.6) in VRF "red"

  A maximum of 128 of these inter-VRF IP DHCP Relay address configurations is allowed per node. A VRF route leak configuration is required for these configurations. In the preceding example, a VRF route leak configuration is required on the default VRF as follows:
  - ip route 100.1.1.2/32 next-hop-vrf blue 100.1.1.2
  - ip route 12.3.4.6/32 next-hop-vrf red 14.3.4.6

VRF configuration examples to avoid

The following examples of VRF configurations are not recommended for IP DHCP Relay.

- The same IP DHCP Relay address configured on different VRFs. As an example:
  - VE interface 100 in default VRF
  - IP address of interface - 3.1.1.1/24
  - IP DHCP Relay address (30.1.1.2) in VRF "blue"
  - IP DHCP Relay address (30.1.1.2) in VRF "red"

- The same IP DHCP Relay address configured on two interfaces with the same address, and both interfaces are on different VRFs. As an example:
  - **VE interface 100 in default VRF**
  - IP address of interface - 3.1.1.1/24
  - IP DHCP Relay address (20.1.1.2) in VRF "blue"
  - **VE interface 200 in VRF "blue"**
  - IP address of interface - 3.1.1.1/24
  - IP DHCP Relay address (20.1.1.2) in VRF "blue"

Displaying IP DHCP Relay statistics

Display information about the DHCP Relay function, such as the DHCP Server IP address configured on the device and the number of DHCP packets received and dropped by the interface configured for IP DHCP Relay.

Use the **show ip dhcp relay statistics** command to display the following information about the IP DHCP Relay function:

- DHCP Server IP Address configured in the device.
- Number of DHCP DISCOVERY, OFFER, REQUEST, ACK, NAK, DECLINE, INFORM, and RELEASE packets received.
- Number of DHCP client packets received (on port 67) and relayed by the Relay Agent.
- Number of DHCP server packets received (on port 68) and relayed by the Relay Agent.
- Number of DHCP client packets dropped by the Relay Agent.
- Number of DHCP server packets dropped by the Relay Agent.

NOTE
You can specify a list of RBridge IDs separated by commas, or a range separated by a dash (for example, 1-2). No spaces are allowed in the range string. The range does not need to be contiguous (for example, 1-2,5). You can also specify “all” for all RBridge IDs in the logical chassis cluster.

1. Access a device where an IP DHCP Relay has been configured on an interface.
2. In privileged EXEC mode, enter **show ip dhcp relay statistics**. Optionally, you can specify specific RBridge IDs if you only want to view the statistics for those RBridges.

The following example displays statistics on a local switch.

```
device# show ip dhcp relay statistics


                DHCP Relay Statistics - RBridge Id: 3
                --------------------------------------
Address    Disc.    Offer   Req.    Ack    Nak    Decline    Release    Inform
-------    -----    -----   ----    ---    ---    -------    -------    ------
10.1.0.1     400      100   2972   2968      0          0          0         0
20.2.0.1     400      100   2979   2975      0          0          0         0
30.3.0.1     400      100   3003   2998      0          0          0         0
40.4.0.1     400      100   3026   3018      0          0          0         0

Client Packets: 12780
Server Packets: 12359
Client Packets Dropped: 0
Server Packets Dropped: 0
```

The following example displays statistics for a cluster with RBridge 1 and RBridge 3.

```
device# show ip dhcp relay statistics rbridge-id 1,3
DHCP Relay Agent Information Option Enabled

                DHCP Relay Statistics - RBridge Id: 1
                ---------------------------------------
Address   Disc.   Offer  Req.   Ack   Nak   Decline   Release   Inform
-------   -----   -----  ----   ---   ---   -------   -------   ------
2.3.4.5     300     100  1211  1201     0         0         0        0
10.0.1.2    300     100  1211  1207     0         0         0        0

Client Packets: 2701
Server Packets: 2932
Client Packets Dropped: 0
Server Packets Dropped: 0

                DHCP Relay Statistics - RBridge Id: 3
                ---------------------------------------
Address   Disc.   Offer  Req.   Ack   Nak   Decline   Release   Inform
-------   -----   -----  ----   ---   ---   -------   -------   ------
10.0.0.5      0       0     0     0     0         0         0        0
10.0.1.2      0       0     0     0     0         0         0        0

Client Packets: 0
Server Packets: 0
Client Packets Dropped: 0
Server Packets Dropped: 0
```

Displaying IP DHCP Relay addresses on specific devices

Use the show ip dhcp relay address command to display all IP DHCP Relay addresses configured on specific switches (specified by RBridge IDs) in a logical chassis cluster. You can use the **all** keyword to display configured addresses on all RBridge IDs in a cluster.

1. Access a device where an IP DHCP Relay has been configured.
2. In privileged EXEC mode, enter the **show ip dhcp relay address rbridge-id** *rbridge-id* command.

The following is an example of displaying addresses configured on interfaces of a local switch. Notice that the RBridge ID is not needed in the command.

```
device# show ip dhcp relay address
DHCP Relay Information Option: Disabled

                              Rbridge Id:    2
                              ----------------
Interface                     Relay Address               VRF Name
---------                     -------------               --------
Te 2/0/35                     10.5.1.1                    default-vrf
Te 2/0/35                     10.3.4.5                    default-vrf
Ve 300                        10.0.1.2                    default-vrf
Ve 300                        10.0.0.5                    default-vrf
```

The following is an example of displaying addresses configured on interfaces on a specific RBridge.

```
device# show ip dhcp relay address rbridge-id 53
DHCP Relay Information Option: Disabled

                            Rbridge Id:   53
                            ----------------
Interface                   Relay Address               VRF Name
---------                   -------------               --------
Ve 300                      20.0.1.2                    blue
Ve 300                      30.1.1.5                    green
Ve 300                      40.2.1.1                    default-vrf
```

The following is an example of displaying addresses configured on interfaces on all RBridge IDs in a logical chassis cluster.

```
device# show ip dhcp relay address rbridge-id all
DHCP Relay Information Option: Disabled

                            Rbridge Id:   53
                            ----------------
Interface                   Relay Address               VRF Name
---------                   -------------               --------
Ve 300                      20.0.1.2                    blue
Ve 300                      30.1.1.5                    green
Ve 300                      40.2.1.1                    default-vrf
DHCP Relay Information Option: Disabled

                            Rbridge Id:    2
                            ----------------
Interface                   Relay Address               VRF Name
---------                   -------------               --------
Te 2/0/35                   10.5.1.1                    default-vrf
Te 2/0/35                   10.3.4.5                    default-vrf
Ve 300                      10.0.1.2                    default-vrf
Ve 300                      10.0.0.5                    default-vrf
```

The following example displays configured IPv4 DHCP Relay addresses on a specified physical interface.

```
device# show ip dhcp relay address interface tengigabitethernet 2/0/37
DHCP Relay Information Option: Disabled
DHCP Remote ID (Type:Length:Vlan:Mac): 00:08:0000:0027f8cad501
Rbridge Id: 2
----------------
Interface Relay Address VRF Name
--------- ------------- --------
Te 2/0/37 8.8.8.8 default-vrf
DHCP relay information option 13
```

The following example displays configured IPv4 DHCP Relay addresses on a specified port channel and RBridge.

```
device# show ip dhcp relay address interface port-channel 10 rbridge-id 2
DHCP Relay Information Option: Disabled
DHCP Remote ID (Type:Length:Vlan:Mac): 00:08:0000:0027f8cad51d
Rbridge Id: 2
----------------
Interface Relay Address VRF Name
--------- ------------- --------
Po 10 20.1.1.1 default-vrf
```

The following example displays configured IPv4 DHCP relay addresses on a VE interface and RBridge.

```
device# show ip dhcp relay address interface ve 10 rbridge-id 2
DHCP Relay Information Option: Disabled
DHCP Remote ID (Type:Length:Vlan:Mac): 00:08:000a:0027f8cad4d9
Rbridge Id: 2
----------------
Interface Relay Address VRF Name
--------- ------------- --------
Ve 10 4.4.4.4 default-vrf
```

The following example displays configured IPv4 DHCP relay addresses on a VE interface and RBridge.

```
device# show ip dhcp relay address interface ve 10 rbridge-id 51,53
DHCP Relay Information Option: Disabled
DHCP Remote ID (Type:Length:Vlan:Mac): 00:08:000a:0027f852c94e
Rbridge Id: 51
----------------
Interface Relay Address VRF Name
--------- ------------- --------
Ve 10 4.4.4.4 default-vrf
DHCP Relay Information Option: Disabled
DHCP Remote ID (Type:Length:Vlan:Mac): 00:08:000a:0027f8cb1377
Rbridge Id: 53
----------------
Interface Relay Address VRF Name
--------- ------------- --------
Ve 10 14.1.1.1 default-vrf
```

Displaying IP DHCP relay addresses for an interface

You can display IP DHCP relay addresses configured on specific interfaces of a device.

To display the IP DHCP relay addresses configured for an interface, use the **show ip dhcp relay address interface** command followed by the interface ID to display IP DHCP relay addresses configured on a specific interface.

1. Access a device where an IP DHCP relay has been configured on an interface.
2. In privileged EXEC mode, enter the **show ip dhcp relay address interface** command followed by the interface ID.

The following is an example for a 10 GbE interface of a local switch.

```
device# show ip dhcp relay address interface te 2/0/35
DHCP Relay Information Option: Disabled
DHCP Remote ID (Type:Length:Vlan:Mac): 00:08:0000:0027f8cad4ff
                            Rbridge Id:    2
                            ----------------
Interface                   Relay Address                VRF Name
---------                   -------------                --------
Te 2/0/35                   10.5.1.1                     default-vrf
Te 2/0/35                   10.3.4.5                     default-vrf
```

The following is an example for a VE interface on a specific switch (RBridge ID 2).

```
device# show ip dhcp relay address int ve 300 rb 2
DHCP Relay Information Option: Disabled
DHCP Remote ID (Type:Length:Vlan:Mac): 00:08:012c:0027f8cad4d9
                            Rbridge Id:    2
                            ----------------
Interface                   Relay Address                VRF Name
---------                   -------------                --------
Ve 300                      10.0.1.2                     default-vrf
Ve 300                      10.0.0.5                     default-vrf
```

The following is an example for displaying addresses on for a specific VE interface on a range of switches (specified by RBridge IDs) in a logical chassis cluster.

NOTE

You can specify a list of RBridge IDs separated by commas, or a range separated by a dash (for example, 1-2). No spaces are allowed in the range string. The range does not need to be contiguous (for example, 1-2,5). You can also specify **all** for all RBridge IDs in the logical chassis cluster.

```
device# show ip dhcp re address int ve 300 rbridge-id 2,53
DHCP Relay Information Option: Disabled
DHCP Remote ID (Type:Length:Vlan:Mac): 00:08:012c:0027f8cad4d9
                              Rbridge Id:    2
                              ----------------
Interface                     Relay Address                VRF Name
---------                     -------------                --------
Ve 300                        10.0.1.2                     default-vrf
Ve 300                        10.0.0.5                     default-vrf
DHCP Relay Information Option: Disabled
DHCP Remote ID (Type:Length:Vlan:Mac): 00:08:012c:0027f8cb1377
                              Rbridge Id:   53
                              ----------------
Interface                     Relay Address                VRF Name
---------                     -------------                --------
Ve 300                        20.0.1.2                     blue
Ve 300                        30.1.1.5                     green
Ve 300                        40.2.1.1                     default-vrf
```

Clearing IP DHCP relay statistics

Use the **clear ip dhcp relay statistics** command to clear all IP DHCP Relay statistics for specific relay IP addresses, for addresses on specific RBridge IDs, or all addresses for RBridge IDs in a logical chassis cluster.

For command parameters you can specify the IP DHCP Relay address and RBridge IDs. You can specify a list of RBridge IDs separated by commas, or a range separated by a dash (for example, 1-2). No spaces are allowed in the range string. The range does not need to be contiguous (for example, 1-2,5). You can also specify "all" for all RBridge IDs in the logical chassis cluster.

1. Access a device where an IP DHCP Relay has been configured on an interface.
2. In privileged EXEC mode, issue the **clear ip dhcp relay statistics ip-address** *ip-address* **rbridge-id** *rbridge-id*.

The following command clears statistics for IP DHCP Relay address 10.1.0.1 configured on RBridges 1, 3, and 5.

```
device# clear ip dhcp relay statistics ip-address 10.1.0.1 rbridge-id 1,3,5
```

The following command clears statistics for all IP DHCP Relay addresses on RBridges 1, 3, and 5.

```
device# clear ip dhcp relay statistics rbridge-id 1,3,5
```

High Availability support

IP DHCP relay address configurations are maintained when control is switched from the active to the standby management module (MM) in the VDX 8770-4 and VDX 8770-8 chassis.

Two management modules (MMs) provide redundancy on the VDX 8770-4 and VDX 8770-8 chassis. These modules host the distributed Network OS that provides overall management for the chassis. If the active module becomes unavailable, the standby module automatically takes over management of the system and becomes the active MM. For more information, refer to the "Configuring High Availability" section of the *Extreme Network OS Management Configuration Guide*.

IP DHCP relay address configurations are maintained on the new active MM and the MM will continue to relay DHCP packets between DHCP clients and servers. IP DHCP relay statistics will not be maintained, however.

DHCPv6

DHCPv6 overview

The Dynamic Host Configuration Protocol for IPv6 (DHCP) enables DHCP servers to pass configuration parameters such as IPv6 network addresses to IPv6 nodes.

The DHCPv6 protocol offers the capability of automatic allocation of reusable network addresses and additional configuration flexibility.

On Network OS devices you can configure the DHCPv6 relay agent.

DHCP relay agent for IPv6

A client locates a DHCPv6 server using a reserved, link-scoped multicast address. Direct communication between the client and server requires that they are attached by the same link. In some situations where ease-of-management, economy, and scalability are concerns, you can allow a DHCPv6 client to send a message to a DHCPv6 server using a DHCPv6 relay agent.

A DHCPv6 relay agent, which may reside on the client link, but is transparent to the client, relays messages between the client and the server. Multiple DHCPv6 relay agents can exist between the client and server. DHCPv6 relay agents can also receive relay-forward messages from other relay agents; these messages are forwarded to the DHCPv6 server specified as the destination.

When the relay agent receives a message, it creates a new relay-forward message, inserts the original DHCPv6 message, and sends the relay-forward message as the DHCPv6 server.

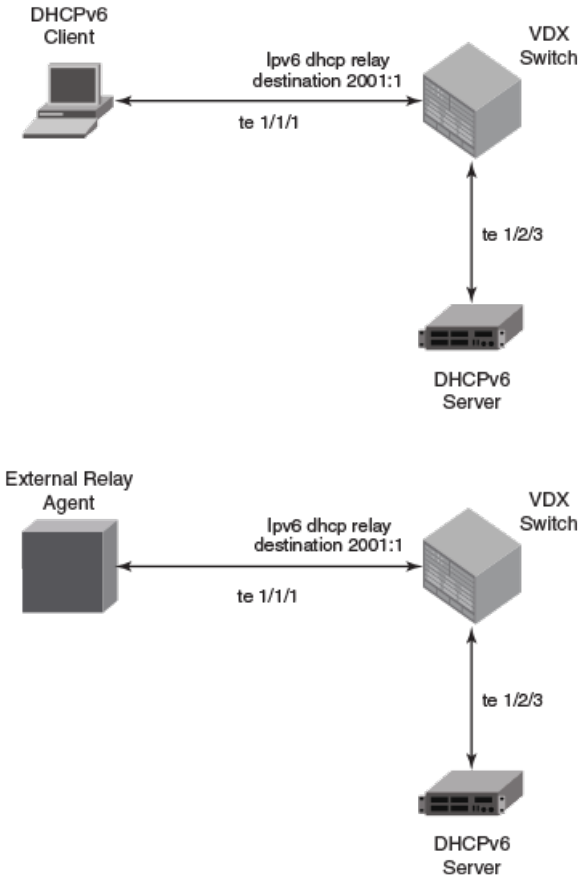
DHCPv6 relay agent

A DHCPv6 relay agent, which may reside on the client's link, is used to relay messages between the client and the server when they are not on the same IPv6 link.

The DHCPv6 relay agent operation is transparent to the client. A DHCPv6 client locates a DHCPv6 server using a reserved, link-scoped multicast address. For direct communication between the DHCPv6 client and the DHCPv6 server, both the client and the server must be attached to the same link. However, in some situations where ease of management, economy, or scalability is a concern, it is desirable to allow a DHCPv6 client to send a message to a DHCPv6 server that is not connected to the same link.

The first figure depicts the relay agent functionality when the relay agent receives DHCP packets from a client on the same link. The second figure depicts the relay agent functionality when the relay agent receives DHCP packets from another relay agent on the same link.

FIGURE 12 DHCPv6 relay functionality



DHCPv6 multicast addresses and UDP ports

The relay agent uses specific multicast addresses and UDP ports for the DHCPv6 functionality.

Multicast addresses

All_DHCP_Relay_Agents_and_Servers (FF02::1:2) is a link-scoped multicast address used by the client to communicate with neighboring (for example, on-link) relay agents and servers. All servers and relay agents are members of this multicast group.

All_DHCP_Servers (FF05::1:3) is a site-scoped multicast address used by the relay agent to communicate with servers, either because the relay agent wants to send messages to all servers or because it does not know the unicast addresses of the servers. To use this address a relay agent must have an address of sufficient scope to be reachable by the servers. All servers within the site are members of this multicast group.

UDP ports

The relay agent listens on UDP ports 546 and 547 for packets sent by clients and servers. The relayed packets use the source port 547.

DHCPv6 address assignment

The DHCPv6 relay agent informs hosts to use one of the following address assignment methods.

Basic DHCPv6 relay assignment

The DHCPv6 Relay agent relays the DHCP messages from clients and other relay agents to a list of destination addresses, which includes unicast IPv6 addresses, the All_DHCP_Servers multicast address, or other user-defined IPv6 multicast group address, on the same VRF as the interface on which the client resides or on a different VRF.

Stateful DHCPv6

DHCPv6 can be used separately, or in addition to stateless address auto-configuration (SLAAC) to obtain configuration parameters. With stateful DHCPv6, the router can signal the attached clients to use DHCPv6 to acquire an IPv6 address by setting the "O" and "M" bits in the Router Advertisement (RA) message.

The following commands set these bits to support stateful DHCPv6:

ipv6 nd managed-config-flag - When set, clients use DHCPv6 for address configuration.

ipv6 nd other-config-flag - When set, hosts use DHCPv6 to obtain other (non-address) information when set. Relay forwarding supports these flags.

Stateless DHCPv6

In stateless DHCPv6, a device uses stateless address auto-configuration (SLAAC) to assign one or more IPv6 addresses to an interface, while it utilizes DHCPv6 to receive additional parameters, such as DNS or NTP server addresses, that may not be available through SLAAC. This is accomplished by setting the "O" bit in the relay agent, using the following CLI:

ipv6 nd other-config-flag - When set, hosts use DHCPv6 to obtain other (non-address) information.

DHCPv6 prefix delegation

The DHCPv6 relay agent relays all the DHCPv6 messages (which may or may not contain the DHCPv6 options) that are to be relayed across the DHCPv6 client and the server. The relay agent will not access or extract the information present in the prefix delegation option.

Relay chaining

DHCPv6 messages can be relayed through multiple relay agents. The Relay-Reply message from the server will be relayed back to the client in the same path it took to reach the server.

Relay-message option

The DHCPV6 relay agent includes the relay message option in all the RELAY-FORW messages.

Remote-ID option

The DHCPv6 relay agent supports the Remote-ID option (option code 37). No user configuration is necessary for this method. The DHCPv6 unique identifier (DUID) of relay agent is used as the remote ID.

Interface-ID option

The DHCPv6 relay agent supports the Interface-ID option to identify the interface on which the client message was received. No user configuration is necessary for this method.

DHCPv6 message format

The DHCPv6 message format varies from the DHCPv4 message format. This section describes the events that occur when a DHCPv6 client sends a Solicit message to locate DHCPv6 servers.

- Solicit (1) - A DHCPv6 client sends a Solicit message to locate DHCPv6 servers.
- Advertise (2) - A server sends an Advertise message to indicate that it is available for DHCP service, in response to a Solicit message received from a client.
- Request (3) - A client sends a Request message to request configuration parameters, including IP addresses or delegated prefixes, from a specific server.
- Confirm (4) - A client sends a Confirm message to any available server to determine whether the addresses it was assigned are still appropriate to the link to which the client is connected. This could happen when the client detects either a link-layer connectivity change or if it is powered on and one or more leases are still valid. The confirm message confirms whether the client is still on the same link or whether if it has been moved. The actual lease(s) are not validated; just the prefix portion of the addresses or delegated prefixes.
- Renew (5) - A client sends a Confirm message to any available server to determine whether the addresses it was assigned are still appropriate to the link to which the client is connected. This could happen when the client detects either a link-layer connectivity change or if it is powered on and one or more leases are still valid. The confirm message confirms whether the client is still on the same link or whether it has been moved. The actual lease(s) are not validated; just the prefix portion of the addresses or delegated prefixes.
- Rebind (6) - A client sends a Rebind message to any available server to extend the lifetimes on the addresses assigned to the client and to update other configuration parameters; this message is sent after a client receives no response to a Renew message.
- Reply (7) - A server sends a Reply message containing assigned addresses and configuration parameters in response to a Solicit, Request, Renew, Rebind message received from a client. A server sends a Reply message containing configuration parameters in response to an Information-request message. A server sends a Reply message in response to a Confirm message confirming or denying that the addresses assigned to the client are appropriate to the link to which the client is connected. A server sends a Reply message to acknowledge receipt of a Release or Decline message.
- Release (8) - A client sends a Release message to the server that assigned addresses to the client to indicate that the client will no longer use one or more of the assigned addresses.
- Decline (9) - A client sends a Decline message to a server to indicate that the client has determined that one or more addresses assigned by the server are already in use on the link to which the client is connected.
- Reconfigure (10) - A server sends a Reconfigure message to a client to inform the client that the server has new or updated configuration parameters, and that the client needs to initiate a Renew/Reply or Information-request/Reply transaction with the server in order to receive the updated information.

- Information-request (11) - A client sends an Information-request message to a server to request configuration parameters without the assignment of any IP addresses to the client.
- Relay-forward (12) - A relay agent sends a Relay-forward message to relay messages to servers, either directly or through another relay agent. The received message, either a client message or a Relay-forward message from another relay agent, is encapsulated in an option in the Relay-forward message.
- Relay-reply - A server sends a Relay-reply message to a relay agent containing a message that the relay agent delivers to a client. The Relay-reply message may be relayed by other relay agents for delivery to the destination.

  NOTE
  The **show ipv6 dhcp relay statistics** command does not display the show packet count in the statistics for DHCPv6 Advertise and Re-configure messages, as it is sent from the DHCPv6 server to DHCPv6 client directly, not through the relay server.

The table lists the DHCPv4 and DHCPv6 message types:

TABLE 11 DHCPv4 message versus DHCPv6 message

DHCPv6 message type	DHCPv4 message type
Solicit (1)	DHCPDISCOVER
Advertise (2)	DHCPOFFER
Request (3), Renew (5), Rebind (6)	DHCPREQUEST
Reply (7)	DHCPPACK/DHCPNAK
Release (8)	DHCPRELEASE
Information-Request (11)	DHCPINFORM
Decline (9)	DHCPDECLINE
Confirm (4)	None
Reconfigure (10)	DHCPFORCERENEW
Relay-Forward(12), Relay-Reply (13)	None

DHCPv6 relay provisioning

Use the scalability numbers in the following table to create and apply the IPv6 DHCP relay configuration on Layer 3 interfaces.

The table lists the scalability numbers for VDX 8770 and VDX 6740:

TABLE 12 Scalability numbers

DHCP configuration	VDX 8770 (per device)	VDX 6740 (per device)
Total Number of DHCPv4 server addresses (**ip dhcp relay address**)	4000	2000
DHCPv4 server addresses per Interface (**ip dhcp relay address**)	16	16
Total number of DHCPv6 server addresses (**ipv6 dhcp relay address**)	4000	2000
DHCPv6 server addresses per interface (**ipv6 dhcp relay address**)	16	16
Number of inter-VRF client-server configurations for v4 or v6 relay (maximum size of the HSL database for each address family)	128	128
Maximum number of client packets (burst rate-limit) - client bindings per second	256 packets per second	1000 packets per second

Configuring IPv6 DHCP relay

Configure the IPv6 DHCP relay agent on any Layer 3 (L3) interface using the IP address of the DHCP server where client requests are to be forwarded.

You can configure the IPv6 DHCP relay agent using the **ipv6 dhcp relay address** command followed by the IP address of the DHCP server. Use the **use-vrf** *vrf-name* parameter if the DHCP server and client interface are on different Virtual Forwarding and Routing (VRF) instances. You can configure up to 16 relay destination addresses per interface.

L3 interfaces can be a virtual Ethernet (VE) or a physical 1, 10, or 40 GbE interface.

The following are the considerations when configuring IPv6 DHCP Relay:

- If the relay address is a link local address or a multicast address, an outgoing interface must be configured for IPv6 relay to function.
- In instances where the server address is relayed to a different VRF compared to client connected interface VRF, in addition to the relay address you must also specify the user-vrf, otherwise IPv6 relay may not function correctly.
- IPv6 route leaking is required for IPv6 reachability.
- Use the following commands in addition to the **ipv6 dhcp relay address** command on the layer 3 interface level so that client goes to the BOUND state.

  For example, a windows device used as a DHCPv6 client looks for these M and O flags in their received router advertisements before initialization.

  device (config)# int te 47/8/10

  device (conf-if-te-47/8/10)# ipv6 nd managed-config-flag (When set, clients use the DHCPv6 protocol for address configuration)

  device (conf-if-te-47/8/10)# ipv6 nd other-config-flag (When set, hosts use DHCPv6 to obtain 'other' non-address information)

Perform the following steps to configure an IPv6 DHCP Relay:

1. In privileged EXEC mode, issue the **configure terminal** command to enter the global configuration mode.

   ```
   device# configure
   ```

2. Enter the **interface** command followed by the interface ID to enter the interface configuration mode where you want to configure the IP DHCP Relay.

   ```
   device(config)# int ve 100
   ```

3. For VE interfaces, enter the **ipv6 dhcp relay address** *ip-addr* **use-vrf** *vrf-name* command where *ip-addr* is the IP address of the DHCP server. Use the **use-vrf** *vrf-name* option if the DHCP server is on a different VRF instance than the interface where the client is connected.

   ```
   (config)# int ve 100
   device(config-Ve-100)# ipv6 dhcp relay address 2001::1122:AABB:CCDD:3344 use-vrf blue
   ```

4. To remove the IP DHCP Relay address enter the **no ipv6 dhcp relay address** *ip-addr* **use-vrf** *vrf-name* command.

The following example specifies the IP DHCP relay address on VE 100 and a ten gigabit ethernet interface.

```
device(config)# int ve 100
device(config-Ve-100)# ipv6 dhcp relay address 2001::1122:AABB:CCDD:3344 use-vrf blue
device(config)# int TenGiga 2/3/1
switch(config-if)# ipv6 dhcp relay address fe80::224:38ff:febb:e3c0  interface tengigabitethernet 2/5
```

Displaying DHCPv6 relay addresses on specific devices

Use the show ipv6 dhcp relay address rbridge-id command to display all IP DHCP Relay addresses configured on specific switches (specified by RBridge IDs) in a logical chassis cluster. You can use the **all** parameter to display configured addresses on all RBridge IDs in a cluster.

1. Access a device where an IP DHCPv6 Relay has been configured.
2. In privileged EXEC mode, execute the **show ipv6 dhcp relay address rbridge-id** *rbridge-id* command.

The following is an example of displaying addresses configured on interfaces of a local switch. Notice that the RBridge ID is not needed in the command.

```
device# show ipv6 dhcp relay address

              RBridge Id:  2
              ----------------
DHCPv6 unique identifier(DUID): 0102111f70027f8cad50d

Interface     Relay Address                            VRF Name                Outgoing Interface
---------     -----------                              --------                ------------------

Te 2/0/35      4001::101                                blue                   Te 2/0/34
Te 2/0/35       fe80::8                                 blue                   Ve 100
Ve 200    5001:1234:1234:2101:1234:1234:3103:1234   default-vrf                Te 2/0/2
```

NOTE

You can specify a list of RBridge IDs separated by commas, or a range separated by a dash (for example, 1-2). No spaces are allowed in the range string. The range does not need to be contiguous (for example, 1-2,5). You can also specify **all** for all RBridge IDs in the logical chassis cluster.

The following is an example of displaying addresses configured on interfaces on a specific RBridge.

```
device# show ipv6 dhcp relay address rbridge-id 2

               RBridge Id: 2
               ----------------
DHCPv6 unique identifier(DUID): 0102111f70027f8cad50d

Interface      Relay Address                              VRF Name        Outgoing Interface
-------------  -------------                              --------        ------------------
Te 2/0/35      4001::101                                   blue            Te 2/0/34

Te 2/0/35      fe80::8                                     blue            Ve 100

Ve 200         5001:1234:1234:2101:1234:1234:3103:1234     default-vrf     Te 2/0/2
```

The following is an example of displaying addresses configured on interfaces on all RBridge IDs in a logical chassis cluster.

```
device# show ipv6 dhcp relay address rbridge-id all

               RBridge Id: 2
               ------------
DHCPv6 unique identifier(DUID): 0102111f70027f8cad50d

Interface      Relay Address                          VRF Name          Outgoing Interface
----------     -------------                          ---------         ------------------
Te 2/0/35       4001::101                             blue               Te 2/0/34

Te 2/0/35       fe80::8                               blue               Ve 100
Ve 200     5001:1234:1234:2101:1234:1234:3103:1234    default-vrf        Te 2/0/2

               RBridge Id: 3
               ------------
DHCPv6 unique identifier(DUID): 0102a1b00027f8cb13ab

Interface      Relay Address                                      VRF Name
----------     -------------                                      ---------
Ve 300         5001:1234:1234:2101:1234:1234:3103:1234            default-vrf
```

Displaying DHCPv6 relay addresses for an interface

You can display IPv6 DHCP Relay addresses configured on a specific interfaces of the device.

To display the IPv6 DHCP Relay addresses configured for an interface, use the **show ipv6 dhcp relay address interface** command followed by the specific interface to display IP DHCP Relay addresses configured on a specific interface.

1. Access a device where an IP DHCP Relay has been configured on an interface.
2. In privileged EXEC mode, execute the **show ip dhcp relay address interface** command followed by the interface ID.

The following is an example for a 10GbE interface of a local switch.

```
device# show ipv6 dhcp relay address interface Te 3/0/21
                                 Rbridge Id:    3
                                  ----------------

DHCPv6 unique identifier(DUID): 0102111f70027f8cad50d

Interface              Relay Address                  VRF Name           Outgoing Interface
---------              -------------                -------------        ----------------------
Te 3/0/21             4001::101                       default-vrf
Te 3/0/21               fe80::8                           blue                        ve 100
```

The following is an example for a Virtual Ethernet interface on a specific switch (RBridge ID 1).

```
device# show ipv6 dhcp rel add int ve 300 rbridge-id 1

              RBridge Id:    1
              ----------------
DHCPv6 unique identifier(DUID): 0102111f70027f8cad50d

Interface     Relay Address       VRF Name      Outgoing Interface
---------     -------------       --------      ------------------
Ve 300        fe80::8           default-vrf
```

The following is an example for displaying addresses on for a specific Virtual Ethernet interface on a range of switches (specified by RBridge IDs) in a logical chassis cluster.

NOTE
You can specify a list of RBridge IDs separated by commas, or a range separated by a dash (for example, 1-2). No spaces are allowed in the range string. The range does not need to be contiguous (for example, 1-2,5). You can also specify "all" for all RBridge IDs in the logical chassis cluster.

```
device# show ipv6 dhcp rel add int ve
300 rbridge-id 1,3

             RBridge Id:    1
             ----------------
DHCPv6 unique identifier(DUID): 0102111f70027f8cad50d

Interface    Relay Address        VRF Name             Outgoing Interface
---------    -------------        --------             ------------------
Ve 300       4001::101             default-vrf
Ve 300       4001::102             default-vrf

             RBridge Id: 3
             -------------
DHCPv6 unique identifier(DUID): 0102a1b00027f8cb13ab

Interface    Relay Address                         VRF Name         Outgoing Interface
---------    -------------                          --------        ------------------
Ve 300       2001::1122:AABB:CCDD:3344              blue
Ve 300       fe80::224:38ff:febb:e3c0               green
Ve 300       2001::1122:AABB:CCDD:3355             default-vrf
```

Displaying IPv6 DHCP relay statistics

Display information about the DHCP Relay function, such as the DHCP Server IP address configured on the device and the number of various DHCP packets received by the interface configured for IP DHCP Relay.

Use the **show ipv6 dhcp relay statistics** command to display the following information about the IPv6 DHCP Relay function:

- Number of SOLICIT, REQUEST, CONFIRM, RENEW, REBIND, RELEASE, DECLINE, INFORMATION-REQUEST, RELAY-FORWARD, RELAY-REPLY packets received.
- Number of RELAY-FORWARD and REPLY packets sent and packets dropped.

NOTE
You can specify a list of RBridge IDs separated by commas, or a range separated by a dash (for example, 1-2). No spaces are allowed in the range string. The range does not need to be contiguous (for example, 1-2,5). You can also specify "all" for all RBridge IDs in the logical chassis cluster.

1. Access a device where an IPv6 DHCP Relay has been configured on an interface.
2. In privileged EXEC mode, enter **show ipv6 dhcp relay statistics**. Optionally, you can specify specific RBridge IDs if you only want to view the statistics for those RBridges.

The following is an example of displaying statistics on a local switch.

```
device# show ipv6 dhcp relay statistics
                              Rbridge Id: 10
Packets dropped : 0
  Error : 0
Packets received : 60
  SOLICIT : 6
  REQUEST : 6
  CONFIRM : 0
  RENEW : 2
  REBIND : 0
  RELEASE : 6
  DECLINE : 0
  INFORMATION-REQUEST : 0
  RELAY-FORWARD : 0
  RELAY-REPLY : 40
Packets sent : 60
  RELAY-FORWARD : 20
  REPLY : 40
```

The following is an example of displaying statistics on RBridge 10.

```
device# show ipv6 dhcp relay statistics rbridge-id 10

                              Rbridge Id: 10
Packets dropped : 0
  Error : 0
Packets received : 60
  SOLICIT : 6
  REQUEST : 6
  CONFIRM : 0
  RENEW : 2
  REBIND : 0
  RELEASE : 6
  DECLINE : 0
  INFORMATION-REQUEST : 0
  RELAY-FORWARD : 0
  RELAY-REPLY : 40
Packets sent : 60
  RELAY-FORWARD : 20
  REPLY : 40
```

Clearing IP DHCPv6 relay statistics

Use the **clear ip dhcpv6 relay statistics** command to clear all IPv6 DHCP Relay statistics for specific relay IP addresses, for addresses on specific RBridge IDs, or all addresses for RBridge IDs in a logical chassis cluster.

1. Access a device where an IPv6 DHCP Relay has been configured on an interface.
2. In privileged EXEC mode, issue the **clear ipv6 dhcp relay statistics ip-address** *ip-address* **rbridge-id** *rbridge-id*.

The following command clears statistics for IPv6 DHCP Relay address 10.1.0.1 configured on RBridges 1, 3, and 5.

```
clear ipv6 dhcp relay statistics ip-address 10.1.0.1 rbridge-id 1,3,5
```

OSPFv2

OSPFv2 overview

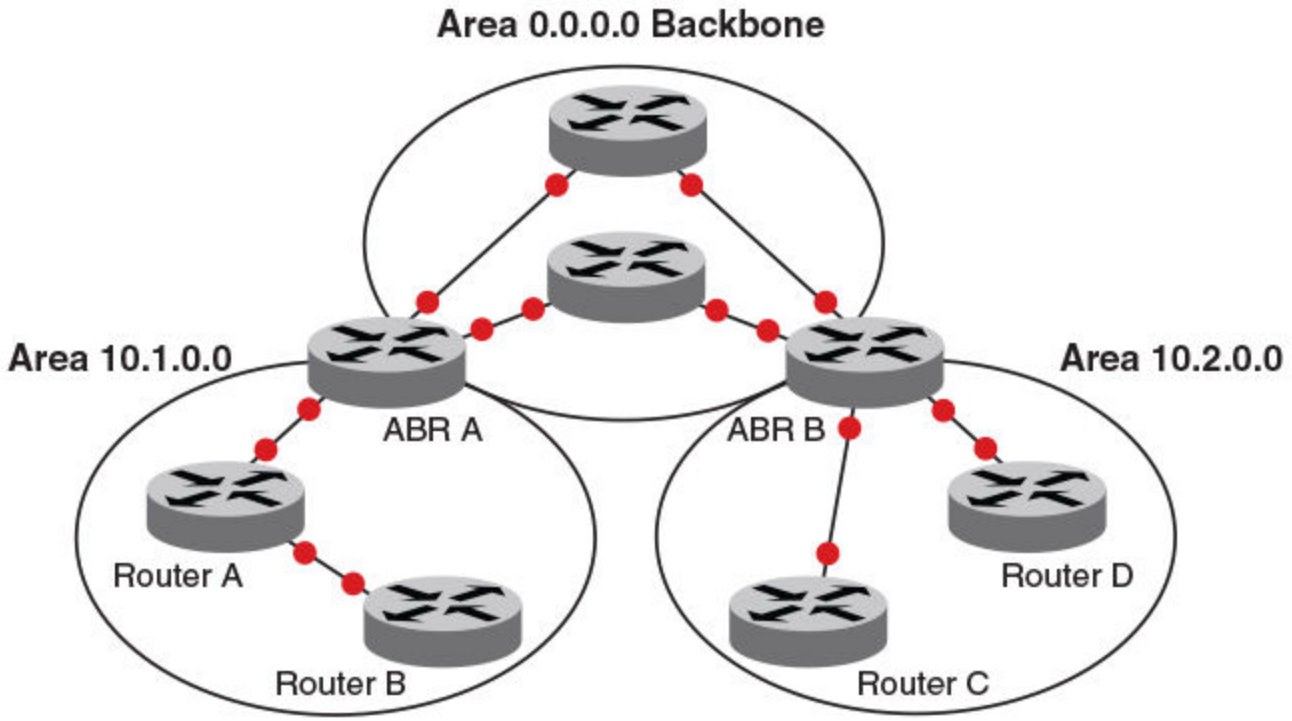
Open Shortest Path First Version 2 (OSPFv2) is a link-state routing protocol that uses link-state advertisements (LSAs) to update neighboring routers about a router's interfaces. Each router maintains an identical area-topology database to determine the shortest path to any neighboring router.

OSPF is built upon a hierarchy of network components and areas. The highest level of the hierarchy is the autonomous system. An autonomous system is defined as a number of networks, all of which share the same routing and administration characteristics. A backbone area forms the core of the network, connecting all other areas. Details of these and other OSPF components are provided below.

Autonomous System

An Autonomous System can be divided into multiple areas. Each area represents a collection of contiguous networks and hosts. Areas limit the amount of advertisements sent within the network. This is known as flooding. An area is represented in OSPFv2 by either an IP address or a number.

FIGURE 13 OSPF operating in a network



NOTE
For details of components and virtual links, refer to OSPFv2 components and roles on page 114 and Virtual links on page 125, respectively.

Once OSPFv2 is enabled on the system, the user assigns an IP address or number as the *area ID* for each area. The area ID is representative of all IP addresses (subnets) on a router port. Each port on a router can support one area.

OSPFv2 components and roles

OSPFv2 can be configured on either a point-to-point or broadcast network.

Devices can take a variety of roles in an OSPFv2 topology, as discussed below.

Area Border Routers

An OSPF router can be a member of multiple areas. Routers with membership in multiple areas are known as Area Border Routers (ABRs). All ABRs must have either a direct or indirect link to an OSPF backbone area (also known as area 0 or area 0.0.0.0). Each ABR maintains a separate topological database for each area the router is in. Each topological database contains all LSA databases for each

router within a given area. The routers within the same area have identical topological databases. An ABR is responsible for forwarding routing information or changes among its border areas.

For more information on OSPFv2 areas, refer to the *OSPFv2 areas* section.

Autonomous System Boundary Routers

An Autonomous System Boundary Router (ASBR) is a router that is running multiple protocols and serves as a gateway to routers outside the OSPF domain and those operating with different protocols. The ASBR is able to import and translate different protocol routes into OSPF through a process known as redistribution.

Designated routers

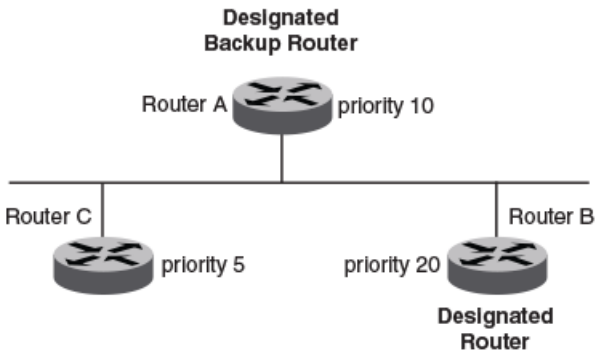
In an OSPF broadcast network, OSPF elects one router to serve as the designated router (DR) and another router on the segment to act as the backup designated router (BDR). This minimizes the amount of repetitive information that is forwarded on the network. OSPF forwards all messages to the designated router.

On broadcast networks such as LAN links, all routers on the LAN other than the DR and BDR form full adjacencies with the DR and BDR and pass LSAs only to them. The DR forwards updates received from one neighbor on the LAN to all other neighbors on that same LAN. One of the main functions of a DR is to ensure that all the routers on the same LAN have identical LSDBs. Therefore, on broadcast networks, an LSDB is synchronized between a DROther (a router that is not a DR or a BDR) and its DR and BDR.

> NOTE
> In an OSPF point-to-point network, where a direct Layer 3 connection exists between a single pair of OSPF routers, there is no need for designated or backup designated routers.

In a network with no designated router and no backup designated router, the neighboring router with the highest priority is elected as the DR, and the router with the next highest priority is elected as the BDR, as shown in the figure below. Priority is a configurable option at the interface level; refer to the **ip ospf priority** command in the *Extreme Network OS Command Reference*.

FIGURE 14 Designated and backup router election



If the DR goes off line, the BDR automatically becomes the DR. The router with the next highest priority becomes the new BDR.

If two neighbors share the same priority, the router with the highest router ID is designated as the DR. The router with the next highest router ID is designated as the BDR. The DR and BDRs are recalculated after the OSPF protocol is disabled and re-enabled by means of the **[no] router ospf** command.

NOTE
By default, the device's router ID is the IP 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 IP address configured on the device.

When multiple routers on the same network are declaring themselves DRs, then both the priority and router ID are used to select the designated router and backup designated routers.

The DR and BDR election process is performed when one of the following events occurs:

- An interface is in a waiting state and the wait time expires.
- An interface is in a waiting state and receives a hello packet that addresses the BDR.
- A change in the neighbor state occurs, such as the following:
  - A neighbor state transitions from ATTEMPT state to a higher state.
  - Communication to a neighbor is lost.
  - A neighbor declares itself to be the DR or BDR for the first time.

Enabling OSPFv2

A number of steps are required when enabling OSPFv2 on a device.

Consider the following when enabling OSPFv2 on a device.

- Redistribution must be enabled on devices configured to operate as ASBRs.
- All device ports must be assigned to one of the defined areas on an OSPF device. When a port is assigned to an area, all corresponding subnets on that port are automatically included in the assignment.

1. Enter the **router ospf** command in RBridge ID configuration mode to enable OSPF on the device.
2. Assign the areas to which the device will be attached.
3. Assign individual interfaces to the OSPF areas.
4. Assign a virtual link to any ABR that does not have a direct link to the OSPF backbone area.
5. Refer to Changing default settings on page 141.

Backbone area

The backbone area (also known as area 0 or area 0.0.0.0) forms the core of OSPF networks. All other areas should be connected to the backbone area either by a direct link or by virtual link configuration. Routers that have interfaces in both backbone area and (at least one) non-backbone area are called Area Border Routers (ABR). Inter area routing happens via ABRs.

The backbone area is the logical and physical structure for the OSPF domain and is attached to all non-zero areas in the OSPF domain.

The backbone area is responsible for distributing routing information between non-backbone areas. The backbone must be contiguous, but it does not need to be physically contiguous; backbone connectivity can be established and maintained through the configuration of virtual links.

Setting up the backbone area

The following example sets up the backbone area.

1. Enter the **configure terminal** command to access global configuration mode.

```
device# configure terminal
```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

```
device(config)# rbridge-id 10
```

3. Enter the **router ospf** command to enter OSPF router configuration mode and enable OSPF on the device.

```
device(onfig-rbridge-id-10)# router ospf
```

4. Enter the **area** command and specify *0.0.0.0* to configure the backbone area. .

```
device(config-router-ospf-vrf-default-vrf)# area 0.0.0.0
```

5. Enter the **exit** command until you return to global configuration mode.

```
device(config-router-ospf-vrf-default-vrf)# exit
```

6. Enter the **interface vlan** command followed by the VLAN number to create a VLAN.

```
device(config)# interface vlan 1001
```

7. Enter the **rbridge-id** command followed by the RBridge ID to enter RBridge configuration mode.

```
device(config)# rbridge-id 10
```

8. Enter the **interface ve** command followed by the VLAN number to enter interface configuration mode.

```
device(config-rbridge-id-10)# interface Ve 1001
```

9. Enter the **ip address** operand followed by the IP address/subnet for the interface.

```
device(config-Ve-1001 # ip address 101.1.1.1/24
```

10. Enter the **ip ospf area** command followed by the area ID to assign the interface to this area.

```
device(config-Ve-1001)# ip ospf area 0.0.0.0
```

```
device# configure terminal
device(config)# rbridge 10
device(config)# ip router-id 10.11.12.13
device(config-rbridge-id-10)# router ospf
device(config-router-ospf-vrf-default-vrf)# area 0.0.0.0
device(config-router-ospf-vrf-default-vrf)# exit
device(config-rbridge-id-10)# exit
device(config)# interface vlan 1001
device(config-Vlan-1001)# rbridge 10
device(config-rbridge-id-10)# interface Ve 1001
device(config-Ve-1001 # ip address 101.1.1.1/24
device(config-Ve-1001)# ip ospf area 0.0.0.0
```

Area range

You can further consolidate routes at an area boundary by defining an area range. The area range allows you to assign an aggregate address to a range of IP and IPv6 addresses.

This aggregate value becomes the address that is advertised instead of all the individual addresses it represents being advertised. Only this aggregate or summary address is advertised into other areas instead of all the individual addresses that fall in the configured range. Area range configuration can considerably reduce the number of Type 3 summary LSAs advertised by a device. You have the option of adding the cost to the summarized route. If you do not specify a value, the cost value is the default range metric calculation for the generated summary LSA cost. You can temporarily pause route summarization from the area by suppressing the type 3 LSA so that the component networks remain hidden from other networks.

You can assign up to 32 ranges in an OSPFv2 area.

Assigning an area range

Ranges for an area can be assigned. Ranges allow a specific IP address and mask to represent a range of IP addresses within an area, so that only that reference range address is advertised to the network, instead of all the addresses within that range. Each area can have up to 32 range addresses.

1. Enter the **configure terminal** command to access global configuration mode.

```
device# configure terminal
```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

```
device(config)# rbridge-id 122
```

3. Enter the **router ospf** command to enter OSPF router configuration mode and enable OSPFv2 on the device.

```
device(config-rbridge-id-122)# router ospf
```

4. Enter the **area range** command, specifying an area ID, and enter the range. Repeat as necessary.

```
device(config-router-ospf-vrf-default-vrf)# area 10.0.0.10 range 10.45.0.0 255.255.0.0
device(config-router-ospf-vrf-default-vrf)# area 10.0.0.20 range 10.45.0.0 255.255.0.0
```

The following example defines an area range for subnets on 10.0.0.10 and 10.0.0.20.

```
device# configure terminal
device(config)# rbridge-id 101
device(config-rbridge-id-101)# router ospf
device(config-router-ospf-vrf-default-vrf)# area 10.0.0.10 range 10.45.0.0 10.255.0.0
device(config-router-ospf-vrf-default-vrf)# area 10.0.0.20 range 10.45.0.0 10.255.0.0
```

Area types

OSPFv2 areas can be normal, a stub area, a totally stubby area (TSA), or a not-so-stubby area (NSSA).

- Normal: OSPFv2 devices within a normal area can send and receive external link-state advertisements (LSAs).
- Stub: OSPFv2 devices within a stub area cannot send or receive external LSAs. In addition, OSPFv2 devices in a stub area must use a default route to the area's Area Border Router (ABR) to send traffic out of the area.

- NSSA: The Autonomous System Boundary Router (ASBR) of an NSSA can import external route information into the area.
  - ASBRs redistribute (import) external routes into the NSSA as type 7 LSAs. Type 7 External LSAs are a special type of LSA generated only by ASBRs within an NSSA, and are flooded to all the routers within only that NSSA.
  - ABRs translate type 7 LSAs into type 5 External LSAs, which can then be flooded throughout the autonomous system. The NSSA translator converts a type 7 LSA to a type 5 LSA if F-bit and P-bit are set and there is a reachable forwarding address. You can configure summary-addresses on the ABR of an NSSA so that the ABR converts multiple type 7 external LSAs received from the NSSA into a single type 5 external LSA.

  When an NSSA contains more than one ABR, OSPFv2 elects one of the ABRs to perform the LSA translation for NSSA. OSPFv2 elects the ABR with the highest router ID. If the elected ABR becomes unavailable, OSPFv2 automatically elects the ABR with the next highest router ID to take over translation of LSAs for the NSSA. The election process for NSSA ABRs is automatic.
- TSA: Similar to a stub area, a TSA does not allow summary routes in addition to not having external routes.

Stub area and totally stubby area

A stub area is an area in which advertisements of external routes are not allowed, reducing the size of the database. A totally stubby area (TSA) is a stub area in which summary link-state advertisement (type 3 LSAs) are not sent. A default summary LSA, with a prefix of 0.0.0.0/0 is originated into the stub area by an ABR, so that devices in the area can forward all traffic for which a specific route is not known, via ABR.

A stub area disables advertisements of external routes. By default, the ABR sends summary LSAs (type 3 LSAs) into stub areas. You can further reduce the number of LSAs sent into a stub area by configuring the device to stop sending type 3 LSAs into the area. You can disable the summary LSAs to create a TSA when you are configuring the stub area or after you have configured the area.

The ABR of a totally stubby area disables origination of summary LSAs into this area, but still accepts summary LSAs from OSPF neighbors and floods them to other neighbors.

When you enter the **area stub** command with the **no-summary** keyword and specify an area to disable the summary LSAs, the change takes effect immediately. If you apply the option to a previously configured area, the device flushes all the summary LSAs it has generated (as an ABR) from the area with the exception of the default summary LSA originated. This default LSA is needed for the internal routers, since external routes are not propagated to them.

NOTE

Stub areas and TSAs apply only when the 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.

Disabling summary LSAs for a stub area

LSAs can be disabled for a stub area.

1. Enter the **configure terminal** command to access global configuration mode.

   ```
   device# configure terminal
   ```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

   ```
   device(config)# rbridge-id 122
   ```

3. Enter the **router ospf** command to enter OSPF router configuration mode and enable OSPFv2 on the device.

```
device(config-rbridge-id-122)# router ospf
```

4. Enter the **area stub** command, specifying an area and a cost, followed by the **no-summary** parameter to set an additional cost on a specified stub area and prevent any Type 3 and Type 4 summary LSAs from being injected into the area.

```
device(config-router-ospf-vrf-default-vrf)# area 40 stub 99 no-summary
```

The following example configures a stub area, specifying a cost of 99 and preventing any Type 3 and Type 4 summary LSAs from being injected into the area.

```
device# configure terminal
device(config)# rbridge-id 101
device(config-rbridge-id-101)# router ospf
device(config-router-ospf-vrf-default-vrf)# area 40 stub 99 no-summary
```

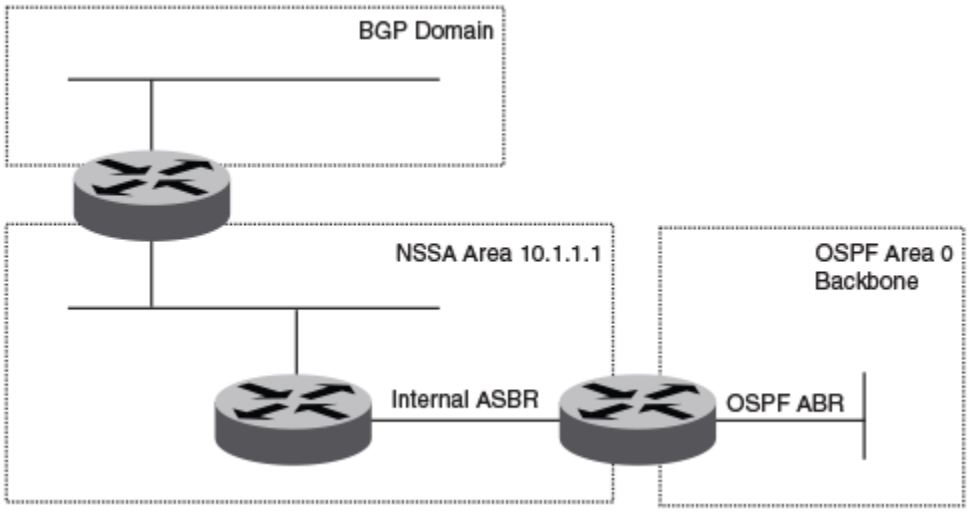
Not-so-stubby area (NSSA)

The OSPFv2 not-so-stubby area (NSSA) enables you to configure OSPFv2 areas that provide the benefits of stub areas, but that also are capable of importing external route information. OSPFv2 does not flood external routes from other areas into an NSSA, but does translate and flood route information from the NSSA into other areas such as the backbone. Since external routes are not published, a Type 7 default LSA with a prefix of ::/0 and a cost of 10 is originated into the NSSA area by the ABR to ensure that traffic passes through.

NSSAs are especially useful when you want to summarize type 5 External LSAs (external routes) before forwarding them into an OSPFv2 area. The OSPFv2 specification prohibits summarization of type 5 LSAs and requires OSPFv2 to flood type 5 LSAs throughout a routing domain. When you configure an NSSA, you can specify a summary-address for aggregating the external routes that the NSSA's ABR exports into other areas.

The figure below shows an example of an OSPFv2 network containing an NSSA.

FIGURE 15 OSPF network containing an NSSA



This example shows two routing domains, a BGP domain and an OSPF domain. The ASBR inside the NSSA imports external routes from BGP into the NSSA as type 7 LSAs, which the ASBR floods throughout the NSSA.

The ABR translates the type 7 LSAs into type 5 LSAs. If a summary-address is configured for the NSSA, the ABR also summarizes the LSAs into an aggregate LSA before flooding the type 5 LSAs into the backbone.

Because the NSSA is partially stubby the ABR does not flood external LSAs from the backbone into the NSSA. To provide access to the rest of the Autonomous System (AS), the ABR generates a default type 7 LSA into the NSSA.

ABRs of an NSSA area can be configured with the no-summary parameter to prevent the generation of type 3 and type 4 summary LSAs into the area. The only exception is the default type 3 LSA, with a prefix of 0.0.0.0/0. The default type 7 LSA is not originated in this case.

Appendix-E implementation for OSPFv2 Type7 LSA

The Link State ID in an NSSA Type 7 LSA is usually set to the described network's IP address. This leads to a limitation that we cannot redistribute multiple networks having the same address, but different masks, since they will result in same Link State ID. Appendix-E clause in RFC 2328 allows a way to overcome this limitation by setting network's host bits in the Link State ID.

When there is a need to originate more than one Type 7 OSPF NSSA LSA with same network address, LSA for the least specific prefix (least prefix length) uses Link State ID same as that of the network address. OSPF NSSA LSAs with more specific prefixes (higher prefix lengths) use Link State ID with their host-bits set.

This feature is always enabled.

The following example displays a static route with LSA for the least specific prefix (least prefix length) for OSPF NSSA LSA configuration.

```
device# show ip ospf database link-state nssa

Area ID         Type LS ID           Adv Rtr         Seq(Hex)        Age  Cksum
1               NSSA 10.0.0.0        47.3.200.1      0x80000001      146  0x2335
  LSA Header:  age: 146, options: 0x0 , seq-nbr: 0x80000001, length: 36
  Network Mask: 255.0.0.0
  TOS 0:  metric_type: 2, metric: 10
          forwarding_address: 100.1.2.3
          external_route_tag: 0


Area ID         Type LS ID           Adv Rtr         Seq(Hex)        Age  Cksum
1               NSSA 10.0.255.255    47.3.200.1      0x80000001      123  0x2335
  LSA Header:  age: 123, options: 0x0 , seq-nbr: 0x80000001, length: 36
  Network Mask: 255.255.0.0
  TOS 0:  metric_type: 2, metric: 10
          forwarding_address: 100.1.2.3
          external_route_tag: 0
device# configure terminal
device(config)# rbridge-id 1
device(config-rbridge-id-1)# ip route 10.0.0.0/16 47.3.1.200
```

Configuring an NSSA

OSPFv2 areas can be defined as NSSA areas with modifiable parameters.

1. Enter the **configure terminal** command to access global configuration mode.

   ```
   device# configure terminal
   ```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

   ```
   device(config)# rbridge-id 122
   ```

3. Enter the **router ospf** command to enter OSPF router configuration mode and enable OSPFv2 on the device.

```
device(config-rbridge-id-122)# router ospf
```

4. Enter the **area nssa** command and specify an area address and a cost.

```
device(config-router-ospf-vrf-default-vrf)# area 10.1.1.1 nssa 1
```

Area 10.1.1.1 is defined as an NSSA.

The following example configures OSPF area 10.1.1.1 as an NSSA.

```
device# configure terminal
device(config)# rbridge-id 122
device(config-rbridge-id-122)# router ospf
device(config-router-ospf-vrf-default-vrf)# area 10.1.1.1 nssa 1
```

Configuring a summary-address for the NSSA

If you want the ABR that connects the NSSA to other areas to summarize the routes in the NSSA before translating them into type 5 LSAs and flooding them into the other areas, configure an address range summary-address. The ABR creates an aggregate value based on the address range. The aggregate value becomes the address that the ABR advertises instead of advertising the individual addresses represented by the aggregate. You can configure up to 32 ranges in an OSPFv2 area.

1. Enter the **configure terminal** command to access global configuration mode.

```
device# configure terminal
```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

```
device(config)# rbridge-id 122
```

3. Enter the **router ospf** command to enter OSPF router configuration mode and enable OSPFv2 on the device.

```
device(config-rbridge-id-122)# router ospf
```

4. Enter the **area nssa** command, specifying an area and a cost.

```
device(config-router-ospf-vrf-default-vrf)# area 10.1.1.1 nssa 10
```

5. Enter the **summary-address** command, followed by the IP address and mask for the summary route.

```
device(config-router-ospf-vrf-default-vrf)# summary-address 10.10.1.0    10.10.2.0
```

The following example configures a summary-address in NSSA 10.1.1.1.

```
device# configure terminal
device(config)# rbridge-id 101
device(config-rbridge-id-101)# router ospf
device(config-router-ospf-vrf-default-vrf)# area 10.1.1.1 nssa 10
device(config-router-ospf-vrf-default-vrf)# summary-address 10.10.1.0    10.10.2.0
```

Configuring the translator role for a NSSA

The translator role by default is candidate. If you want the ABR to unconditionally assume the role of a NSSA translator, configure the **translator-always** option.

1. Enter the **configure terminal** command to access global configuration mode.

```
device# configure terminal
```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

```
device(config)# rbridge-id 122
```

3. Enter the **router ospf** command to enter OSPF router configuration mode and enable OSPFv2 on the device.

```
device(config-rbridge-id-122)# router ospf
```

4. Enter the **area nssa** command, specifying an area. Use the **translator-always** parameter to configure the translator role, causing the router to unconditionally assume the role of a NSSA translator.

```
device(config-router-ospf-vrf-default-vrf)# area 1.1.1.1 nssa translator-always
```

The following example configures a summary-address in NSSA 1.1.1.1 and configures the router to unconditionally assume the role of a NSSA translator.

```
device# configure terminal
device(config)# rbridge-id 101
device(config-rbridge-id-101)# router ospf
device(config-router-ospf-vrf-default-vrf)# area 1.1.1.1 nssa translator-always
```

Configuring the OSPF default metric

The OSPF default metric and the metric type of default route originated into NSSA area can be specified.

1. Enter the **configure terminal** command to access global configuration mode.

```
device# configure terminal
```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

```
device(config)# rbridge-id 122
```

3. Enter the **router ospf** command to enter OSPF router configuration mode and enable OSPFv2 on the device.

```
device(config-rbridge-id-122)# router ospf
```

4. Enter the **area nssa** command, specifying an area. Use the **default-information-metric** keyword with the **metric-type type-1** keywords to specify that the metric of a neighbor is the cost between itself and the router plus the cost of using this router for routing to the rest of the world.

```
device(config-router-ospf-vrf-default-vrf)# area 10.1.1.1 default-information-metric metric-type
type1
```

The following example specifies that the metric type of the default route originated into NSSA area 10.1.1.1 is type 1.

```
device# configure terminal
device(config)# rbridge-id 101
device(config-rbridge-id-101)# router ospf
device(config-router-ospf-vrf-default-vrf)# area 10.1.1.1 default-information-metric metric-type type1
```

Preventing an NSSA ABR from generating external LSAs into a NSSA area

If you want an ASBR to generate type-5 LSA into normal areas and to prevent it from generating type-7 LSA into the NSSA area, you can configure the no redistribution option.

1. Enter the **configure terminal** command to access global configuration mode.

```
device# configure terminal
```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

```
device(config)# rbridge-id 122
```

3. Enter the **router ospf** command to enter OSPF router configuration mode and enable OSPFv2 on the device.

```
device(config-rbridge-id-122)# router ospf
```

4. Enter the **area nssa** command, specifying an area and a cost. Use the **no-redistribution** parameter to prevent an NSSA ABR from generating external LSAs into a NSSA area.

```
device(config-router-ospf-vrf-default-vrf)# area 6 nssa 4 no-redistribution
```

The following example prevents an NSSA ABR from generating external LSAs into a NSSA area.

```
device# configure terminal
device(config)# rbridge-id 101
device(config-rbridge-id-101)# router ospf
device(config-router-ospf-vrf-default-vrf)# area 6 nssa 4 no-redistribution
```

Assigning interfaces to an area

Once you define OSPFv2 areas, you can assign interfaces to the areas. All device ports must be assigned to one of the defined areas on an OSPF device. When a port is assigned to an area, all corresponding subnets on that port are automatically included in the assignment.

To assign a tengigabitethernet interface 101/0/1 to an area with the IP address of 10.5.0.0, perform the following task:

1. Enter the **configure terminal** command to access global configuration mode.

```
device# configure terminal
```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

```
device(config)# rbridge-id 101
```

3. Enter the **interface** command and specify an interface.

```
device(config-rbridge-id-101)# interface tengigabitethernet 101/0/1
```

4. Issue the **ip ospf area** command followed by the IP address of the area.

```
device(conf-if-te-101/0/1)# ip ospf area 10.5.0.0
```

   If you want to set an interface to passive mode, use the **ip ospf passive** command. If you want to block flooding of outbound LSAs on specific OSPFv2 interfaces, use the **ip ospf database-filter all out** command.(Refer to the *Network OS Command Reference* for details.)

The following example assigns a tengigabitethernet interface to an area with the IP address of 10.5.0.0.

```
device# configure terminal
device(config)# rbridge-id 101
device(config-rbridge-id-101)# interface tengigabitethernet 101/0/1
device(conf-if-te-101/0/1)# ip ospf area 10.5.0.0
```

Link state advertisements

Communication among areas is provided by means of link state advertisements (LSAs). The LSAs supported for each area type are as follows:

- Backbone (area 0) supports LSAs 1, 2, 3, 4, 5, and 7.
- Nonbackbone, supports LSAs 1, 2, 3, 4, and 5.
- Stub area supports LSAs 1, 2, and 3.
- Totally stubby area (TSA) supports LSAs 1 and 2, and also supports a single LSA 3 per ABR, advertising a default route.
- No so stubby area (NSSA) supports LSAs 1, 2, 3, and 7.

Virtual links

All ABRs must have either a direct or indirect link to the OSPFv2 backbone area (0.0.0.0 or 0). If an ABR does not have a physical link to the area backbone, the ABR can configure a virtual link to another router within the same area, which has a physical connection to the area backbone.

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

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

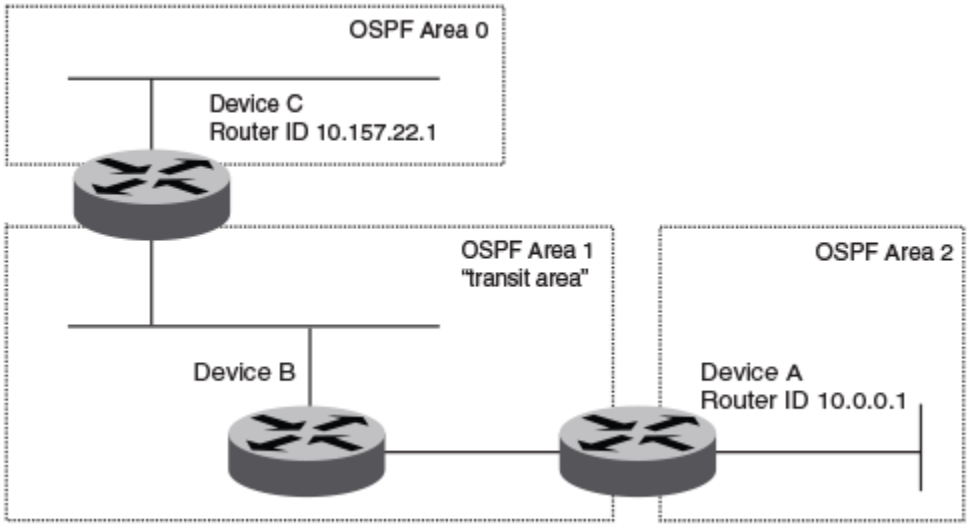
- The transit area ID represents the shared area of the two ABRs and serves as the connection point between the two routers. This number should match the area ID value.
- The neighbor router field is the router ID (IP address) of the router that is physically connected to the backbone, when assigned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

  NOTE
  By default, a device's router ID is the IP 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 IP address configured on the device. When you establish an area virtual link, you must configure it on both of the routers (both ends of the virtual link).

Virtual links cannot be configured in stub areas and NSSAs.

The following figure shows an OSPF area border router, Device A, that is cut off from the backbone area (area 0). To provide backbone access to Device A, you can add a virtual link between Device A and Device C 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.

FIGURE 16 Defining OSPF virtual links within a network



Configuring virtual links

If an Area Border Router (ABR) does not have a physical link to a backbone area, a virtual link can be configured between that ABR and another device within the same area that has a physical link to a backbone area.

A virtual link is configured, and a virtual link endpoint on two devices, ABR1 and ABR2, is defined.

1. On ABR1, enter the **configure terminal** command to access global configuration mode.

   ```
   device# configure terminal
   ```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

   ```
   device(config)# rbridge-id 122
   ```

3. Enter the **router ospf** command to enter OSPF router configuration mode and enable OSPFv2 on the device.

   ```
   device(config-rbridge-id-122)# router ospf
   ```

4. Enter the **area** command to assign an OSPFv2 area ID.

   ```
   device(config-router-ospf-vrf-default-vrf)# area 0
   ```

5. Enter the **area** command to assign an OSPFv2 area ID.

   ```
   device(config-router-ospf-vrf-default-vrf)# area 1
   ```

6. Enter the **area virtual-link** command and the ID of the OSPFv2 device at the remote end of the virtual link to configure the virtual link endpoint.

   ```
   device(config-router-ospf-vrf-default-vrf)# area 1 virtual-link 10.2.2.2
   ```

7. On ABR2, enter the **configure terminal** command to access global configuration mode.

   ```
   device# configure terminal
   ```

8. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

```
device(config)# rbridge-id 104
```

9. Enter the **router ospf** command to enter OSPFv2 router configuration mode and enable OSPFv2 on the device.

```
device(config-rbridge-id-104)# router ospf
```

10. Enter the **area** command to assign an OSPFv2 area ID.

```
device(config-router-ospf-vrf-default-vrf)# area 1
```

11. Enter the **area** command to assign an OSPFv2 area ID.

```
device(config-router-ospf-vrf-default-vrf)# area 2
```

12. Enter the **area virtual-link** command and the ID of the OSPFv2 device at the remote end of the virtual link to configure the virtual link endpoint.

```
device(config-router-ospf-vrf-default-vrf)# area 1 virtual-link 10.1.1.1
```

The following example configures a virtual link between two devices.

```
ABR1:
device1# configure terminal
device1(config)# rbridge-id 122
device1(config-rbridge-id-122)# router ospf
device1(config-router-ospf-vrf-default-vrf)# area 0
device1(config-router-ospf-vrf-default-vrf)# area 1
device1(config-router-ospf-vrf-default-vrf)# area 1 virtual-link 10.2.2.2

ABR2:
device2# configure terminal
device2(config)# rbridge-id 104
device2(config-rbridge-id-104)# router ospf
device2(config-router-ospf-vrf-default-vrf)# area 1
device2(config-router-ospf-vrf-default-vrf)# area 2
device2(config-router-ospf-vrf-default-vrf)# area 1 virtual-link 10.1.1.1
```

Default route origination

When the device is an OSPFv2 Autonomous System Boundary Router (ASBR), you can configure it to automatically generate a default external route into an OSPFv2 routing domain.

By default, a device does not advertise the default route into the OSPFv2 domain. If you want the device to advertise the OSPFv2 default route, you must explicitly enable default route origination. When you enable OSPFv2 default route origination, the device advertises a type 5 default route that is flooded throughout the autonomous system, with the exception of stub areas.

The device advertises the default route into OSPFv2 even if OSPFv2 route redistribution is not enabled, and even if the default route is learned through an iIBGP neighbor when default-information-originate is configured. The device does not, however, originate the default route if the active default route is learned from an OSPFv2 device in the same domain.

> NOTE
> The device does not advertise the OSPFv2 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 OSPFv2 devices are not affected. If you re-enable the default route origination, the change takes effect immediately and you do not need to reload the software.

External route summarization

An ASBR can be configured to advertise one external route as an aggregate for all redistributed routes that are covered by a specified address range.

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

If a route that falls within a configured summary address range is imported by the 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 summary address range is removed by the device, no action is taken if there are other imported routes that fall within the same summary address range; otherwise, the aggregate route is flushed.

You can configure up to 32 summary address ranges. The device sets the forwarding address of the aggregate route to 0 and sets the tag to 0. If you delete a summary 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 (LSDB) overflow condition occurs, all aggregate routes and other external routes are flushed out of the autonomous system. When the device exits the external LSDB overflow condition, all the imported routes are summarized according to the configured summary address ranges.

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

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

NOTE
Only imported, type 5 external LSA routes are affected. A single type 5 LSA is generated and flooded throughout the autonomous system for multiple external routes.

Modifying Shortest Path First timers

The Shortest Path First (SPF) throttle timers can be modified.

1. Enter the **configure terminal** command to access global configuration mode.

```
device# configure terminal
```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode..

```
device(config)# rbridge-id 122
```

3. Enter the **router ospf** command to enter OSPF router configuration mode and enable OSPFv2 globally.

```
device(config-rbridge-id-122)# router ospf
```

4. Enter the **timers** command with the **throttle spf** keyword and specify the SPF delay, the hold time, and the maximum wait time.

```
device(config-router-ospf-vrf-default-vrf)# timers throttle spf 100 500 5000
```

The following example sets the SPF initial delay to 100 milliseconds, the hold time to 500 milliseconds, and the maximum wait time to 5000 milliseconds.

```
device# configure terminal
device(config)# rbridge-id 122
device(config-rbridge-id-122)# router ospf
device(config-router-ospf-vrf-default-vrf)# timers throttle spf 100 500 5000
```

OSPFv2 administrative distance

Devices can learn about networks from various protocols and select a route based on the source of the route information. This decision can be influenced if the default administrative distance for OSPFv2 routes is changed. Consequently, the routes to a network may differ depending on the protocol from which the routes were learned.

You can influence the device's decision by changing the default administrative distance for OSPFv2 routes. You can configure a unique administrative distance for each type of OSPFv2 route. For example, you can configure the Extreme device to prefer a static route over an OSPFv2 inter-area route and to prefer OSPFv2 intra-area routes over 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 OSPFv2 route types:

- External routes
- Intra-area routes
- Inter-area routes
- Route maps

NOTE
The choice of routes within OSPFv2 is not influenced. For example, an OSPFv2 intra-area route is always preferred over an OSPFv2 inter-area route, even if the intra-area route's distance is greater than the inter-area route's distance.

OSPFv2 LSA refreshes

To prevent a refresh from being performed each time an individual LSA's refresh timer expires, OSPFv2 LSA refreshes are delayed for a specified time interval. This pacing interval can be altered.

The device paces OSPFv2 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 device refreshes and sends out together in one or more packets.

The pacing interval, which is the interval at which the device refreshes an accumulated group of LSAs, is configurable in a range from 10 through 1800 seconds (30 minutes). The default is 240 seconds (4 minutes). Thus, every four minutes, the device refreshes the group of accumulated LSAs and sends the group together in the same packets.

The pacing interval is inversely proportional to the number of LSAs the 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 to 100 LSAs), increasing the pacing interval to 10 to 20 minutes may enhance performance only slightly.

Configuring the OSPFv2 LSA pacing interval

The interval between OSPFv2 LSA refreshes can be modified.

1. Enter the **configure terminal** command to access global configuration mode.

   ```
   device# configure terminal
   ```

2. Enter the **rbridge-id** command with an RBridge ID t o enter RBridge ID configuration mode.

   ```
   device(config)# rbridge-id 122
   ```

3. Enter the **router ospf** command to enter OSPF router configuration mode and enable OSPFv2 globally.

   ```
   device(config-rbridge-id-122)# router ospf
   ```

4. Enter the **timers** command with the **lsa-group-pacing** parameter.

   ```
   device(config-router-ospf-vrf-default-vrf)# timers lsa-group-pacing 120
   ```

   The OSPFv2 LSA pacing interval is changed to 120 seconds (2 minutes).

The following example changes the OSPFv2 LSA pacing interval is changed to 120 seconds (2 minutes).

```
device# configure terminal
device(config)# rbridge-id 122
device(config-rbridge-id-122)# router ospf
device(config-router-ospf-vrf-default-vrf)# timers lsa-group-pacing 120
```

OSPFv2 graceful restart

The graceful restart (GR) feature provides a routing device with the capability to inform its neighbors when it is performing a restart.

Neighboring devices, known as GR helpers, are informed via protocol extensions that the device is undergoing a restart and assist in the restart. For the duration of the graceful restart, the restarting device and its neighbors continue forwarding packets ensuring there is no disruption to network performance or topology. Disruptions in forwarding are minimized and route flapping diminished. When the restart is complete, the device is able to quickly resume full operation due to the assistance of the GR helpers. The adjacent devices then return to normal operation.

There are two types of OSPFv2 graceful restart:

- **Planned restart**: the restarting routing device informs its neighbors before performing the restart. The GR helpers act as if the routing device is still within the network topology, continuing to forward traffic to the restarting routing device. A defined interval, known as a “grace period” is set to specify when the neighbors should consider the restart complete and the restarting routing device as part of the network topology again.
- **Unplanned restart**: the routing device restarts without warning due to a software fault.

  NOTE
  In order for a graceful restart on a routing device to be successful, the OSPFv2 neighbors must have GR-helper mode enabled. GR-helper mode is enabled by default.

Disabling OSPFv2 graceful restart

OSPFv2 graceful restart (GR) is enabled by default, and can be disabled on a routing device.

1. Enter the **configure terminal** command to access global configuration mode.

```
device# configure terminal
```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

```
device(config)# rbridge-id 122
```

3. Enter the **router ospf** command to enter OSPF router configuration mode and enable OSPFv2 globally.

```
device(config-rbridge-id-122)# router ospf
```

4. Enter the **no graceful restart** command to disable GR on the device.

```
device(config-router-ospf-vrf-default-vrf)# no graceful-restart
```

The following example disables GR.

```
device# configure terminal
device(config)# rbridge-id 122
device(config-rbridge-id-122)# router ospf
device(config-router-ospf-vrf-default-vrf)# no graceful-restart
```

Re-enabling OSPFv2 graceful restart

If you disable OSPFv2 graceful restart (GR), you can re-enable it. You can also change the maximum restart wait time from the default value of 120 seconds.

NOTE
GR is mutually exclusive to NSR.

1. Enter the **configure terminal** command to access global configuration mode.

```
device# configure terminal
```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

```
device(config)# rbridge-id 122
```

3. Enter the **router ospf** command to enter OSPF router configuration mode and enable OSPFv2 globally.

```
device(config-rbridge-id-122)# router ospf
```

4. Enter the **graceful restart** command to re-enable GR on the device.

```
device(config-router-ospf-vrf-default-vrf)# graceful-restart
```

5. Enter the **graceful restart** command with the **restart-time** parameter and specify a value to change the maximum restart wait time from the default value of 120 seconds.

```
device(config-router-ospf-vrf-default-vrf)# graceful-restart restart-time 240
```

The following example re-enables GR and changes the maximum restart wait time from the default value of 120 seconds to 240 seconds.

```
device# configure terminal
device(config)# rbridge-id 122
device(config-rbridge-id-122)# router ospf
device(config-router-ospf-vrf-default-vrf)# graceful-restart
device(config-router-ospf-vrf-default-vrf)# graceful-restart restart-time 240
```

Disabling OSPFv2 graceful restart helper

The OSPFv2 graceful restart (GR) helper is enabled by default, and can be disabled on a routing device.

1. Enter the **configure terminal** command to access global configuration mode.

   ```
   device# configure terminal
   ```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

   ```
   device(config)# rbridge-id 122
   ```

3. Enter the **router ospf** command to enter OSPF router configuration mode and enable OSPFv2 on the device.

   ```
   device(config-rbridge-id-122)# router ospf
   ```

4. Enter the **graceful-restart** command using the **helper-disable** keyword to disable the GR helper.

   ```
   device(config-router-ospf-vrf-default-vrf)# graceful-restart helper-disable
   ```

The following example disables the GR helper.

```
device# configure terminal
device(config)# rbridge-id 122
device(config-rbridge-id-122)# router ospf
device(config-router-ospf-vrf-default-vrf)# graceful-restart helper-disable
```

OSPFv2 non-stop routing

OSPFv2 can continue operation without interruption during hitless failover when the OSPFv2 non-stop routing (NSR) feature is enabled.

During graceful restart (GR), the restarting neighbors must help build routing information during a failover. However, GR may not be supported by all devices in a network. NSR eliminates this dependency.

NSR does not require support from neighboring devices to perform hitless failover, and OSPF can continue operation without interruption.

NOTE
NSR does not support IPv6-over-IPv4 tunnel and virtual links, so traffic loss is expected while performing hitless failover.

NOTE
NSR and Graceful Restart (GR) are mutually exclusive.

Enabling OSPFv2 NSR

OSPFv2 non-stop routing (NSR) can be re-enabled if it has been disabled. The following task re-enables NSR for OSPFv2.

NOTE
GR is mutually exclusive to NSR.

1. Enter the **configure terminal** command to access global configuration mode.

```
device# configure terminal
```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

```
device(config)# rbridge-id 122
```

3. Enter the **router ospf** command to enter OSPF router configuration mode and enable OSPFv2 globally.

```
device(config-rbridge-id-122)# router ospf
```

4. Enter the **nonstop-routing** command to re-enable NSR on the device.

```
device(config-router-ospf-vrf-default-vrf)# nonstop-routing
```

The following example re-enables NSR for OSPFv2.

```
device# configure terminal
device(config)# rbridge-id 122
device(config-rbridge-id-122)# router ospf
device(config-router-ospf-vrf-default-vrf)# nonstop-routing
```

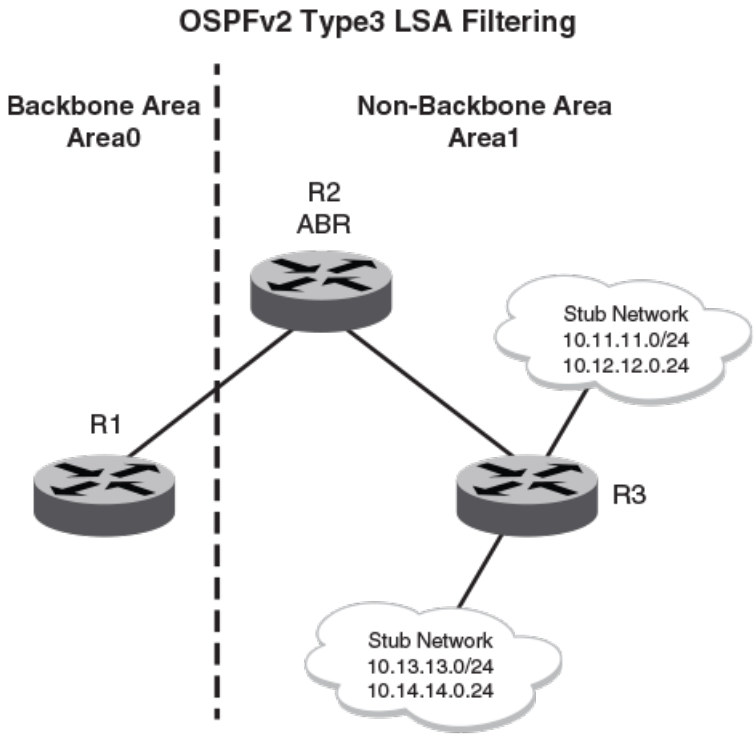
OSPFv2 type 3 LSA filtering

OSPFv2 type 3 LSA filtering provides an ABR that is running the OSPFv2 protocol with the ability to filter type 3 link-state advertisements (LSAs) that are sent between different OSPFv2 areas. Filtering of routes can be defined using prefix-list filters to either permit or deny certain prefixes. Only specified prefixes can be sent from one area to another area and all other prefixes are prohibited. OSPFv2 type 3 LSA filtering can be applied for the LSAs coming into a specific OSPFv2 area or going out of a specific OSPFv2 area, or into and out of the same OSPFv2 areas concurrently. Any change in the prefix-list used for type 3 filtering may result in the advertisement of new summary LSAs or the withdrawal of previously advertised summary LSAs.

Type 3 LSAs refer to summary links and are sent by ABRs to advertise destinations outside the area. OSPFv2 type 3 LSA filtering gives the administrator improved control of route distribution between OSPFv2 areas.

In certain situations, reachability for some IP network prefixes from outside of an area or to a specific area should be restricted. Such a situation is illustrated in the figure below where a device called R3 is advertising stub networks that belong to the non-backbone area (Area1). An ABR, R2, will generate summary routes for these prefixes. If OSPFv2 type 3 LSA filtering is not configured, all the stub networks are seen as summary LSAs in the backbone area (Area 0) and are flooded to all applicable OSPFv2 devices. These type 3 LSAs can be filtered out using the OSPFv2 type3 LSA Filter.

FIGURE 17 OSPFv2 type 3 LSA filtering



Usage and configuration guidelines

- OSPFv2 type 3 LSA filtering is only applicable to ABRs. Configurations are accepted prior to the device becoming an ABR but OSPFv2 type 3 LSA filtering only occurs once the device becomes an ABR.
- When OSPFv2 type 3 LSA filtering is enabled in the “in” direction, all type 3 LSAs originated by the ABR to this area are filtered by the prefix list, based on information from all other areas. Type 3 LSAs, originated as a result of the **area range** command in a different area, are treated like any other individuallly originated type 3 LSA. Any prefix that does not match an entry in the prefix list is implicitly denied.
- When OSPFv2 type 3 LSA filtering is enabled in the “out” direction, all type 3 LSAs advertised by the ABR are filtered by the prefix list, based on information from this area to all other areas. If the **area range** command has been used to configure type 3 LSAs for this area, these type 3 LSAs that correspond to these configurations are treated like any other type 3 LSA. Prefixes that do not match are implicitly denied.
- Type 3 LSAs received from other devices are not filtered out. The OSPFv2 type 3 LSA filter is only applied when the ABR generates summary routes to advertise.
- If a prefix-list used in type 3 LSA filtering is not created or defined, all summary LSAs are discarded.
- OSPFv2 type 3 LSA filtering is supported for both default VRFs and non-default VRFs.

Verifying OSPFv2 type 3 LSA filtering configurations

Use this example procedure to verify OSPFv2 type 3 LSA filtering configurations.

1. Verify OSPFv2 configurations at any level of the CLI.

```
device# show ip ospf database link-state

Link States

Index Area ID         Type LS ID           Adv Rtr         Seq(Hex)        Age  Cksum
1     0               Rtr  192.0.0.1       192.0.0.1       0x80000002      404  0x60b6
2     0               Rtr  1.1.1.1         1.1.1.1         0x80000002      403  0xa9e8
3     0               Net  30.30.30.1      1.1.1.1         0x80000001      403  0x959f
4     0               Summ 10.14.14.0      1.1.1.1         0x80000001      211  0x9b95
5     0               Summ 10.13.13.0      1.1.1.1         0x80000001      237  0xb280
6     0               Summ 10.12.12.0      1.1.1.1         0x80000001      264  0xc96b
7     0               Summ 10.11.11.0      1.1.1.1         0x80000001      542  0xe056
8     0               Summ 20.20.20.0      1.1.1.1         0x80000001      542  0x8497
9     1               Rtr  3.3.3.3         3.3.3.3         0x80000006      212  0x7a6e
10    1               Rtr  1.1.1.1         1.1.1.1         0x80000003      542  0x5775
11    1               Net  20.20.20.2      3.3.3.3         0x80000001      1728 0xd827
12    1               Summ 30.30.30.0      1.1.1.1         0x80000001      542  0x1be2
```

   Information about all summary LSAs is displayed.

2. Configure type 3 LSA filtering on the device.

```
device# configure terminal
device(config)# rbridge-id 122
device(config-rbridge-id-122)# router ospf
device(config-router-ospf-vrf-default-vrf)# area 0 prefix-list AREA_0_IN in
device(config-router-ospf-vrf-default-vrf)# exit
device(config-rbridge-id-122)# ip prefix-list AREA_0_IN  seq 20 deny 10.12.12.0/24
device(config-rbridge-id-122)# ip prefix-list AREA_0_IN  seq 30 permit 10.14.14.0/24
```

   The following filters are applied:

   - Area 0: Inbound filtering permits 10.14.14.0/24 and denies 10.12.12.0/24.4

3. Verify OSPFv2 type 3 LSA filtering configurations using the **show ip ospf database link-state** command.

```
device# show ip ospf database link-state

Link States

Index Area ID         Type LS ID           Adv Rtr         Seq(Hex)        Age  Cksum
1     0               Rtr  192.0.0.1       192.0.0.1       0x80000002      1478 0x60b6
2     0               Rtr  1.1.1.1         1.1.1.1         0x80000002      1477 0xa9e8
3     0               Net  30.30.30.1      1.1.1.1         0x80000001      1477 0x959f
4     0               Summ 10.14.14.0      1.1.1.1         0x80000001      1285 0x9b95
5     1               Rtr  3.3.3.3         3.3.3.3         0x80000006      1286 0x7a6e
6     1               Rtr  1.1.1.1         1.1.1.1         0x80000003      1616 0x5775
7     1               Net  20.20.20.2      3.3.3.3         0x80000002      938  0xd628
8     1               Summ 30.30.30.0      1.1.1.1         0x80000001      1616 0x1be2
```

4. Verify the settings for type 3 LSA filtering in the in direction for OSPFv2 area 0 using the **show ip ospf filtered-lsa area** command.

```
device# show ip ospf filtered-lsa area 0 in

Prefix List Name: AREA_0_IN
Direction: IN
Area: 0
Filtered LSA list:
 Prefix          Mask
10.11.11.0      255.255.255.0
10.12.12.0      255.255.255.0
10.13.13.0      255.255.255.0
20.20.20.0      255.255.255.0
Total number of Filtered LSA :4
```

5. Verify information about type 3 LSA filters configured for OSPFv2 areas using the **show ip ospf config** command.

```
device# show ip ospf config

Router OSPF: Enabled
Nonstop Routing: Disabled
Graceful Restart: Enabled
Graceful Restart Helper: Enabled
Graceful Restart Time: 120

Redistribution: Disabled
Default OSPF Metric: 10
Maximum Paths: 8
OSPF Auto-cost Reference Bandwidth: Disabled
Default Passive Interface: Disabled
OSPF Redistribution Metric: Type2
OSPF External LSA Limit: 14913080
OSPF Database Overflow Interval: 0
RFC 1583 Compatibility: Enabled
VRF Lite capability: Disabled
Router id: 1.1.1.1
OSPF Area currently defined:
Area-ID          Area-Type Cost
1                normal    0
0                normal    0

OSPF type3 filters currently defined:
Area-ID          IN prefix                       OUT prefix
0                AREA_0_IN
```

OSPFv2 over VRF

With Network OS 4.0 and later, OSPFv2 can run over multiple Virtual Routing and Forwarding (VRF) instances. All OSPFv2 commands supported in Network OS 4.0 and later are available over default and non-default OSPF instances.

OSPFv2 maintains multiple instances of the routing protocol to exchange route information among various VRF instances. A multi-VRF-capable device maps an input interface to a unique VRF, based on user configuration. These input interfaces can be physical or a virtual interface. By default, all input interfaces are attached to the default VRF instance.

Multi-VRF for OSPF (also known as VRF-Lite for OSPF) provides a reliable mechanism for trusted VPNs to be built over a shared infrastructure. The ability to maintain multiple virtual routing or forwarding tables allows overlapping private IP addresses to be maintained across VPNs.

Enabling OSPFv2 in a non-default VRF

When OSPFv2 is enabled in a non-default VRF instance, the device enters OSPF router VRF configuration mode. Several commands can then be accessed that allow the configuration of OSPFv2.

A non-default VRF instance has been configured.

1. Enter the **configure terminal** command to access global configuration mode.

```
device# configure terminal
```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

```
device(config)# rbridge-id 122
```

3. Enter the **router ospf** command and specify a VRF name to enter OSPF router VRF configuration mode and enable OSPFv2 on a non-default VRF.

```
device(config-rbridge-id-122)# router ospf vrf green
```

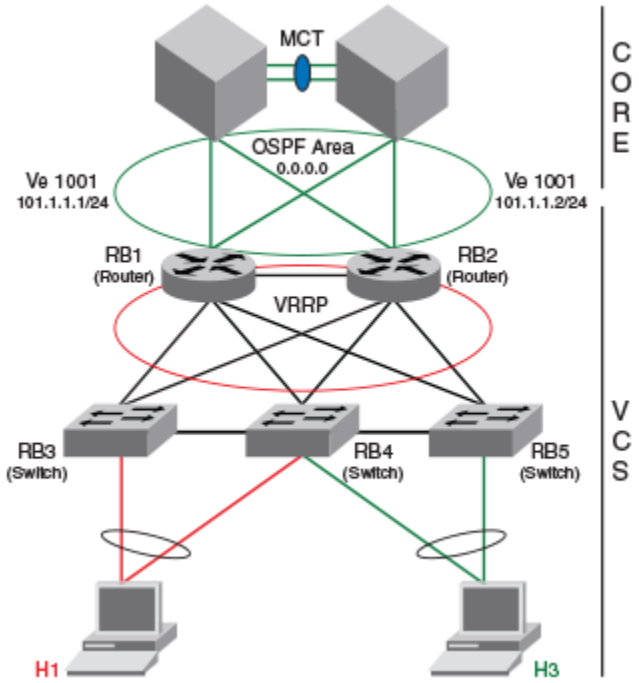
The following example enables OSPFv2 in a non-default VRF.

```
device# configure terminal
device(config)# rbridge-id 122
device(config-rbridge-id-122)# router ospf vrf green
device(config-router-ospf-vrf-green)#
```

OSPFv2 in a VCS environment

The figure below shows one way in which OSPFv2 can be used in a VCS Fabric environment. Routers RB1 and RB2, as well as the devices that aggregate traffic to the core, are configured with OSPFv2. Switches RB3, RB4, and RB5 are Layer 2 switches.

FIGURE 18 OSPF example in a VCS environment



OSPFv2 considerations and limitations

- OSPFv2 must be configured in a Virtual Cluster Switching (VCS) environment.
- The following Extreme platforms support OSPF:
  - VDX 6740
  - VDX 6740T
  - VDX 6740T-1G
  - VDX 6940-36Q
  - VDX 6940-144S
  - VDX 8770-4
  - VDX 8770-8
- OSPFv2 can be configured on either a point-to-point or broadcast network.
- OSPF is supported on unnumbered IP interfaces. Refer to the *Unnumbered IP interfaces* section for more information.
- OSPFv2 can be enabled on the following interfaces: gigabitethernet, tengigabitethernet, fortygigabitethernet, loopback, and ve.
- On enabling OSPFv2 over a loopback interface, the network is advertised as a stub network in the router LSA for the attached area. OSPFv2 control packets, such as *hellos*, are not transmitted on loopback interfaces and adjacencies will not form.
- For VXLAN, if you are configuring a loopback interface to serve as a VTEP, you must manually configure distinct router-ids, using the **ip router id command**, for use by routing protocols.

Configuring the OSPFv2 Max-Metric Router LSA

By configuring the OSPFv2 max-metric router LSA you can enable OSPFv2 to advertise its locally generated router LSAs with a maximum metric.

NOTE
You can configure OSPFv2 max-metric router LSA in either startup or non-startup mode. When you configure max-metric in non-startup mode, it only applies once and is not persistent across reloads or after the **clear ip ospf all** command is issued.

1. Enter the **configure terminal** command to access global configuration mode.

```
device# configure terminal
```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

```
device(config)# rbridge-id 122
```

3. Enter the **router ospf** command to enter OSPF router configuration mode and enable OSPFv2 on the device.

```
device(config-rbridge-id-122)# router ospf
```

4. Enter the **max-metric router-lsa** command with the **all-lsas** parameter to set the set the summary-lsa and external-lsa parameters to the corresponding default max-metric value. .

```
device(config-router-ospf-vrf-default-vrf)# max-metric router-lsa all-lsas
```

The following example configures an OSPFv2 device to advertise the maximum metric value using the **all-lsas** parameter.

```
device# configure terminal
device(config)# rbridge-id 122
device(config-rbridge-id-122)# router ospf
device(config-router-ospf-vrf-default-vrf)# max-metric router-lsa all-lsas
```

Re-enabling OSPFv2 compatibility with RFC 1583

OSPFv2 is compatible with RFC 1583 and maintains a single best route to an autonomous system (AS) boundary router in the OSPF routing table. Disabling this compatibility causes the OSPF routing table to maintain multiple intra-AS paths, which helps prevent routing loops. You can re-enable OSPFv2 compatibility with RFC 1583 if it has been disabled.

1. Enter the **configure terminal** command to access global configuration mode.

```
device# configure terminal
```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

```
device(config)# rbridge-id 122
```

3. Enter the **ip router ospf** command to enter OSPF router configuration mode and enable OSPFv2 on the device.

```
device(config-rbridge-id-122)# router ospf
```

4. Enter the **rfc1583-compatibility** command to re-enable OSPFv2 compatibility with RFC 1583.

```
device(config-router-ospf-vrf-default-vrf)# rfc1583-compatibility
```

The following example re-enables OSPFv2 compatibility with RFC 1583.

```
device# configure terminal
device(config)# rbridge-id 122
device(config-rbridge-id-122)# router ospf
device(config-router-ospf-vrf-default-vrf)# rfc1583-compatibility
```

Enabling OSPFv2 in a VCS environment

Do the following to enable OSPFv2 in a VCS environment.

NOTE

If no RBridge ID is configured on the device, deleting a VE interface will cause a spike in CPU usage. To prevent this, configure an RBridge ID before deleting a VE interface.

1. On device RB1, do the following:
   a) Enter the **configure terminal** command to enter global configuration mode.
   b) Enter the **interface vlan** command followed by the VLAN number to create a VLAN for the router.
   c) Enter the **exit** command to exit interface configuration mode.
   d) Enter the **rbridge-id** command followed by the RBridge ID to enter RBridge configuration mode.
   e) Enter the **router ospf** command to enable the OSPFv2 routing protocol and to enter OSPF router configuration mode.
   f) Enter the **area** command followed by the area ID to create this OSPF area on this router.
   g) Enter the **exit** command to exit OSPF router configuration mode.
   h) Enter the **interface ve** command followed by the VLAN number to enter interface configuration mode.
   i) Enter the **ip address** command followed by the IP address/subnet of the interface.
   j) Enter the **ip ospf area** command followed by the area ID to assign the interface to this area.
   k) Enter the **no shutdown** command.

```
RB1# configure terminal
RB1(config)# interface vlan 1001
RB1(config-Vlan-1001)# exit
RB1(config)# rbridge-id 1

RB1(config-rbridge-id-1)# router ospf
RB1(config-router-ospf-vrf-default-vrf)# area 0.0.0.0
RB1(config-router-ospf-vrf-default-vrf)# exit
RB1(config-rbridge-id-1)# interface ve 1001
RB1(config-Ve-1001)# ip address 101.1.1.1/24
RB1(config-Ve-1001)# ip ospf area 0.0.0.0
RB1(config-Ve-1001)# no shutdown
```

2. On device RB2, do the following:
   a) Enter the **configure terminal** command to enter global configuration mode.
   b) Enter the **interface vlan** command followed by the VLAN number to create a VLAN for the router.
   c) Enter the **exit** command to exit interface configuration mode.
   d) Enter the **rbridge-id** command followed by the RBridge ID to enter RBridge configuration mode.
   e) Enter the **router ospf** command to enable the OSPFv2 routing protocol and to enter OSPF router configuration mode.
   f) Enter the **area** command followed by the area ID to create this OSPF area on this router.
   g) Enter the **exit** command to exit OSPF VRF router configuration mode.
   h) Enter the **interface ve** command followed by the VLAN number to enter interface configuration mode.
   i) Enter the **ip address** command followed by the IP address/subnet of the interface.
   j) Enter the **ip ospf area** command followed by the area ID to assign the interface to this area.
   k) Enter the **no shutdown** command.

```
RB2# configure terminal
RB2(config)# interface vlan 1001
RB2(config-Vlan-1001)# exit
RB2(config)# rbridge-id 2

RB2(config-rbridge-id-2)# router ospf
RB2(config-router-ospf-vrf-default-vrf)# area 0.0.0.0
RB2(config-router-ospf-vrf-default-vrf)# exit
RB2(config-rbridge-id-2)# interface ve 1001
RB2(config-Ve-1001)# ip address 101.1.1.2/24
RB2(config-Ve-1001)# ip ospf area 0.0.0.0
RB2(config-Ve-1001)# no shutdown
```

   l) Assign VLAN 1001 to a vLAG.

Changing default settings

Refer to the relevant Command Reference for other commands you can use to change default OSPF settings. Some commonly configured items include the following:

- Changing reference bandwidth to change interface costs by using the **auto-cost reference-bandwidth** command.
- Defining redistribution filters for the Autonomous System Boundary Router (ASBR) by using the **redistribute** command.

Disabling and re-enabling OSPFv2 event logging

OSPFv2 event logging can be configured, disabled, and re-enabled.

1. Enter the **configure terminal** command to access global configuration mode.

```
device# configure terminal
```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

```
device(config)# rbridge-id 1
```

3. Enter the **router ospf** command to enter OSPF router configuration mode and enable OSPFv2 globally.

```
device(config-rbridge-id-1)# router ospf
```

4. Enter the **no log all** command to disable the logging of all OSPFv2 events.

```
device(config-router-ospf-vrf-default-vrf)# no log all
```

The following example re-enables the logging of all OSPFv2 events.

```
device# configure terminal
device(config)# rbridge-id 1
device(config-rbridge-id-1)# router ospf
device(config-router-ospf-vrf-default-vrf)# log all
```

Understanding the effects of disabling OSPFv2

Consider the following before disabling OSPFv2 on a device:

- If you disable OSPFv2, the device removes all the configuration information for the disabled protocol from the running configuration. Moreover, when you save the configuration to the running configuration file after disabling one of these protocols, all the configuration information for the disabled protocol is removed from the running configuration file.
- If you are testing an OSPFv2 configuration and are likely to disable and re-enable the protocol, you might want to make a backup copy of the running configuration 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 configuration file into the flash memory.
- If the management default route information is available in the Chassis ID (CID) card, the OSPFv2 default route is overwritten by the management default route when the switch reboots. In order to prevent this, remove the management default route after the switch reboots. The OSPFv2 default route is automatically re-instated. Refer to the "Using the Chassis ID (CID) Recovery Tool "chapter in the *Network OS Software Troubleshooting Guide*.

Disabling OSPFv2

To disable OSPFv2 on a device, use the **no router ospf** command:

1. Enter the **configure terminal** command to access global configuration mode.

```
device# configure terminal
```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

```
device(config)# rbridge-id 122
```

3. Enter the **no router ospf** command to disable OSPFv2 on the device.

```
device(config-rbridge-id-122)# no router ospf
```

The following example disables OSPFv2 on a device.

```
device# configure terminal
device(config)# rbridge-id 101
device(config-rbridge-id-101)# no router ospf
```

OSPFv3

OSPFv3 overview

Open Shortest Path First (OSPF) is a link-state routing protocol. Each OSPF device originates link-state advertisement (LSA) packets to describe its link information. These LSAs are flooded throughout the OSPF area. The flooding algorithm ensures that every device in the area has an identical database. Each device in the area then calculates a Shortest Path Tree (SPT) that shows the shortest distance to every other device in the area, using the topology information in the Link State database..

IPv6 supports OSPF Version 3 (OSPFv3), which functions similarly to OSPFv2, the version that IPv4 supports, except for the following enhancements:

- Support for IPv6 addresses and prefixes.
- Ability to configure several IPv6 addresses on a device interface. (While OSPFv2 runs per IP subnet, OSPFv3 runs per link. In general, you can configure several IPv6 addresses on a router interface, but OSPFv3 forms one adjacency per interface only, using the link local address of the interface as the source for OSPF protocol packets. On virtual links, OSPFv3 uses the global IP address as the source. OSPFv3 imports all or none of the address prefixes configured on a router interface. You cannot select the addresses to import.)
- Ability to run one instance of OSPFv2 and one instance of OSPFv3 concurrently on a link.
- Support for IPv6 link-state advertisements (LSAs).

NOTE
Although OSPFv2 and OSPFv3 function in a similar manner, Extreme has implemented the user interface for each version independently of the other. Therefore, any configuration of OSPFv2 features will not affect the configuration of OSPFv3 features and vice versa.

OSPFv3 considerations and limitations

There are a number of things to consider when configuring OSPFv3.

- OSPFv3 must be configured in a VCS environment.
- OSPFv3 can be configured on either a point-to-point or broadcast network.
- OSPFv3 can be enabled on the following interfaces: gigabitethernet, tengigabitethernet, fortygigabitethernet, hundredgigabitethenet, loopback, and ve.
- On enabling OSPFv3 over a loopback interface, the network is advertised as a stub network in the router LSA for the attached area. OSPFv3 control packets, such as *hellos* , are not transmitted on loopback interfaces and adjacencies will not form.

Configuring the router ID

When configuring OSPFv3, the router ID for a device must be specified.

1. Enter the **configure terminal** command to access global configuration mode.

```
device# configure terminal
```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

```
device(config)# rbridge-id 122
```

3. Enter the **ip router-id** command to specify the router ID.

```
device(config-rbridge-id-122)# ip router-id 10.11.12.13
```

The following example configures the router ID for a device.

```
device# configure terminal
device(config)# rbridge-id 122
device(config-rbridge-id-122)# ip router-id 10.11.12.13
```

Enabling OSPFv3

When OSPFv3 is enabled on a device, the device enters OSPFv3 router configuration mode. Several commands can then be accessed that allow the configuration of OSPFv3.

1. Enter the **configure terminal** command to access global configuration mode.

```
device# configure terminal
```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

```
device(config)# rbridge-id 122
```

3. Enter the **ip router-id** command to specify the router ID.

```
device(config-rbridge-id-122)# ip router-id 10.11.12.13
```

4. Enter the **ipv6 router ospf** command to enter OSPFv3 router configuration mode and enable OSPFv3 on the device.

```
device(config-rbridge-id-122)# ipv6 router ospf
```

The following example enables OSPFv3 on a device.

```
device# configure terminal
device(config)# rbridge-id 122
device(config-rbridge-id-122)# ip router-id 10.11.12.13
device(config-rbridge-id-122)# ipv6 router ospf
device(config-ipv6-router-ospf-vrf-default-vrf)#
```

Configuring OSPFv3

A number of steps are required when configuring OSPFv3:

- Configure the router ID.
- Enable OSPFv3 globally.
- Assign OSPFv3 areas.
- Assign OSPFv3 areas to interfaces.

OSPFv3 areas

After OSPFv3 is enabled, you can assign OSPFv3 areas. You can specify the area id in plain number format , such as "area 1", or in ipv4 address format, such as 10.1.1.1. Each device interface can support one area.

NOTE
You can assign only one area on a device interface.

NOTE
You are required to configure a router ID when running only IPv6 routing protocols.

NOTE
By default, the 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 highest-numbered IPv4 address configured on the device. You can also configure router id using the **ip router-id** command.

Backbone area

The backbone area (also known as area 0 or area 0.0.0.0) forms the core of OSPF networks. All other areas should be connected to the backbone area either by a direct link or by virtual link configuration. Routers that have interfaces in both backbone area and (at least one) non-backbone area are called Area Border Routers (ABR). Inter area routing happens via ABRs.

The backbone area is the logical and physical structure for the OSPF domain and is attached to all non-zero areas in the OSPF domain.

The backbone area is responsible for distributing routing information between non-backbone areas. The backbone must be contiguous, but it does not need to be physically contiguous; backbone connectivity can be established and maintained through the configuration of virtual links.

Area range

You can further consolidate routes at an area boundary by defining an area range. The area range allows you to assign an aggregate address to a range of IP and IPv6 addresses.

This aggregate value becomes the address that is advertised instead of all the individual addresses it represents being advertised. Only this aggregate or summary address is advertised into other areas instead of all the individual addresses that fall in the configured range. Area range configuration can considerably reduce the number of Type 3 summary LSAs advertised by a device. You have the option of adding the cost to the summarized route. If you do not specify a value, the cost value is the default range metric calculation for the generated summary LSA cost. You can temporarily pause route summarization from the area by suppressing the type 3 LSA so that the component networks remain hidden from other networks.

You can assign up to 4 ranges in an OSPFv3 area.

Area types

OSPFv3 areas can be normal, a stub area, a totally stubby area (TSA), or a not-so-stubby area (NSSA).

- Normal: OSPFv3 devices within a normal area can send and receive external link-state advertisements (LSAs).
- Stub: OSPFv3 devices within a stub area cannot send or receive External LSAs. In addition, OSPF devices in a stub area must use a default route to the area's Area Border Router (ABR) to send traffic out of the area.
- TSA: A form of stub area, where Type 3 summary routes are also not propagated in addition to Type 5 external routes.
- NSSA: A form of stub area, where Type 5 external routes by Autonomous System Boundary Routers (ASBRs) outside this area are not propagated, but where it is allowed to have an ASBR in the area, that can advertise external information.
  - ASBRs redistribute (import) external routes into the NSSA as type 7 LSAs. Type 7 External LSAs are a special type of LSA generated only by ASBRs within an NSSA, and are flooded to all the routers within only that NSSA.
  - One of the ABRs of the NSSA area is selected as a NSSA translator, and this router translates the area-specific Type 7 LSAs to Type 5 external LSAs which can be flooded throughout the Autonomous System (except NSSA and stub areas).

When an NSSA contains more than one ABR, OSPFv3 elects one of the ABRs to perform the LSA translation for NSSA. OSPF elects the ABR with the highest router ID. If the elected ABR becomes unavailable, OSPFv3 automatically elects the ABR with the next highest router ID to take over translation of LSAs for the NSSA. The election process for NSSA ABRs is automatic.

Assigning OSPFv3 areas

Areas can be assigned as OSPFv3 areas.

Enable IPv6 on each interface on which you plan to enable OSPFv3. You enable IPv6 on an interface by configuring an IP address or explicitly enabling IPv6 on that interface.

1. Enter the **configure terminal** command to access global configuration mode.

   ```
   device# configure terminal
   ```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

   ```
   device(config)# rbridge-id 122
   ```

3. Enter the **ip router-id** command to specify the router ID.

```
device(config-rbridge-id-122)# ip router-id 10.11.12.13
```

4. Enter the **ipv6 router ospf** command to enter OSPFv3 router configuration mode and enable OSPFv3 on the device.

```
device(config-rbridge-id-122)# ipv6 router ospf
```

5. Enter the **area** command to define an OSPFv3 area ID.

```
device(config-ipv6-router-ospf-vrf-default-vrf)# area 0
```

6. Enter the **area** command to define a second OSPFv3 area ID.

```
device(config-ipv6-router-ospf-vrf-default-vrf)# area 10.1.1.1
```

The following example assigns an OSPFv3 ID to two areas. One of the areas is assigned by decimal number. The second area is assigned by IP address.

```
device# configure terminal
device(config)# rbridge-id 122
device(config-rbridge-id-122)# ip router-id 10.11.12.13
device(config-rbridge-id-122)# ipv6 router ospf
device(config-ipv6-router-ospf-vrf-default-vrf)# area 0
device(config-ipv6-router-ospf-vrf-default-vrf)# area 10.1.1.1
```

Assigning OSPFv3 areas to interfaces

Defined OSPFv3 areas can be assigned to device interfaces.

Ensure that OSPFv3 areas are assigned.

> NOTE
> All device interfaces must be assigned to one of the defined areas on an OSPFv3 device. When an interface is assigned to an area, all corresponding subnets on that interface are automatically included in the assignment.

1. Enter the **configure terminal** command to access global configuration mode.

```
device# configure terminal
```

2. Enter the **interface** command and specify an interface.

```
device(config)# interface tengigabitethernet 1/0/1
```

3. Enter the **ipv6 address** command to add an IPv6 address to the interface.

```
device(config-if-te-1/0/1)# ipv6 address 2001:1:0::1:1/64
```

4. Enter the **ipv6 ospf area** command.

```
device(config-if-te-1/0/1)# ipv6 ospf area 0
```

   Area 0 is assigned to the specified interface with the IPv6 address of 2001:1:0:1::1/64.

5. Enter the **exit** command to return to global configuration mode.

```
device(config-if-te-1/0/1)# exit
```

6. Enter the **interface** command and specify an interface.

```
device(config)# interface tengigabitethernet 1/0/2
```

7. Enter the **ipv6 address** command to add an IPv6 address to the interface.

```
device(config-if-te-1/0/2)# ipv6 address 2001:1:0::2:1/64
```

8. Enter the **ipv6 ospf area** command.

```
device(config-if-te-1/0/2)# ipv6 ospf area 1
```

Area 1 is assigned to the specified interface with the IPv6 address of 2001:1:0:2::1/64.

The following example configures and enables OSPFv3 on two specified interfaces, and assigns an interface to two router areas.

```
device# configure terminal
device(config)# interface tengigabitethernet 1/0/1
device(config-if-te-1/0/1)# ipv6 address 2001:1:0::1:1/64
device(config-if-te-1/0/1)# ipv6 ospf area 0
device(config-if-te-1/0/1)# exit
device(config)# interface tengigabitethernet 1/0/2
device(config-if-te-1/0/2)# ipv6 address 2001:1:0::2:1/64
device(config-if-te-1/0/2)# ipv6 ospf area 1
```

Stub area and totally stubby area

A stub area is an area in which advertisements of external routes are not allowed, reducing the size of the database. A totally stubby area (TSA) is a stub area in which summary link-state advertisement (type 3 LSAs) are not sent. A default summary LSA, with a prefix of 0.0.0.0/0 is originated into the stub area by an ABR, so that devices in the area can forward all traffic for which a specific route is not known, via ABR.

A stub area disables advertisements of external routes. By default, the ABR sends summary LSAs (type 3 LSAs) into stub areas. You can further reduce the number of LSAs sent into a stub area by configuring the device to stop sending type 3 LSAs into the area. You can disable the summary LSAs to create a TSA when you are configuring the stub area or after you have configured the area.

The ABR of a totally stubby area disables origination of summary LSAs into this area, but still accepts summary LSAs from OSPF neighbors and floods them to other neighbors.

When you enter the **area stub** command with the **no-summary** keyword and specify an area to disable the summary LSAs, the change takes effect immediately. If you apply the option to a previously configured area, the device flushes all the summary LSAs it has generated (as an ABR) from the area with the exception of the default summary LSA originated. This default LSA is needed for the internal routers, since external routes are not propagated to them.

NOTE

Stub areas and TSAs apply only when the 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.

Configuring a stub area

OSPFv3 areas can be defined as stub areas with modifiable parameters.

1. Enter the **configure terminal** command to access global configuration mode.

```
device# configure terminal
```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

```
device(config)# rbridge-id 122
```

3. Enter the **ip router-id** command to specify the router ID.

```
device(config-rbridge-id-122)# ip router-id 10.4.4.4
```

4. Enter the **ipv6 router ospf** command to enter OSPFv3 router configuration mode and enable OSPFv3 on the device.

```
device(config-rbridge-id-122)# ipv6 router ospf
```

5. Enter the **area stub** command and specify a metric value.

```
device(config-ipv6-router-ospf-vrf-default-vrf)# area 4 stub 100
```

   Area 4 is defined as a stub area with an additional cost of 100.

The following example sets an additional cost of 100 on a stub area defined as 4.

```
device# configure terminal
device(config)# rbridge-id 122
device(config-rbridge-id-122)# ip router-id 10.4.4.4
device(config-rbridge-id-122)# ipv6 router ospf
device(config-ipv6-router-ospf-vrf-default-vrf)# area 4 stub 100
```

Not-so-stubby area

A not-so-stubby-area (NSSA) is an OSPFv3 area that provides the benefits of stub areas with the extra capability of importing external route information. OSPFv3 does not flood external routes from other areas into an NSSA, but does translate and flood route information from the NSSA into other areas such as the backbone.

NSSAs are especially useful when you want to aggregate type 5 External LSAs (external routes) before forwarding them into an OSPFv3 area. When you configure an NSSA, you can specify an address range for aggregating the external routes that the ABR of the NSSAs exports into other areas.

If the router is an ABR, you can prevent any type 3 and type 4 LSA from being injected into the area by configuring a nssa with the **no-summary** parameter. The only exception is that a default route is injected into the NSSA by the ABR, and strictly as a type 3 LSA. The default type 7 LSA is not originated in this case.

By default, the device's NSSA translator role is set to candidate and the router participates in NSSA translation election, if it is an ABR. You can also configure the NSSA translator role.

In the case where an NSSA ABR is also an ASBR, the default behavior is that it originates type 5 LSAs into normal areas and type 7 LSAs into an NSSA. But you can prevent an NSSA ABR from generating type 7 LSAs into an NSSA by configuring the **no-redistribution** parameter.

Configuring an NSSA

OSPFv3 areas can be defined as NSSA areas with configurable parameters.

1. Enter the **configure terminal** command to access global configuration mode.

```
device# configure terminal
```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

```
device(config)# rbridge-id 122
```

3. Enter the **ip router-id** command to specify the router ID.

```
device(config-rbridge-id-122)# ip router-id 10.3.3.3
```

4. Enter the **ipv6 router ospf** command to enter OSPFv3 router configuration mode and enable OSPFv3 on the device.

```
device(config-rbridge-id-122)# ipv6 router ospf
```

5. Enter the **area nssa** command with the **default-information-originate** keyword and specify a cost.

```
device(config-ipv6-router-ospf-vrf-default-vrf)# area 3 nssa default-information-originate metric 33
```

   Area 3 is defined as an NSSA with the default route option and an additional cost of 33.

The following example sets an additional cost of 33 on an NSSA defined as 3.

```
device# configure terminal
device(config)# rbridge-id 122
device(config-rbridge-id-122)# ip router-id 10.3.3.3
device(config-rbridge-id-122)# ipv6 router ospf
device(config-ipv6-router-ospf-vrf-default-vrf)# area 3 nssa default-information-originate metric 33
```

LSA types for OSPFv3

Communication among OSPFv3 areas is provided by means of link-state advertisements (LSAs). OSPFv3 supports a number of 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)
- NSSA External LSAs (Type 7)
- Link LSAs (Type 8)
- Intra-area-prefix LSAs (Type 9)

For more information about these LSAs, refer to RFC 5340.

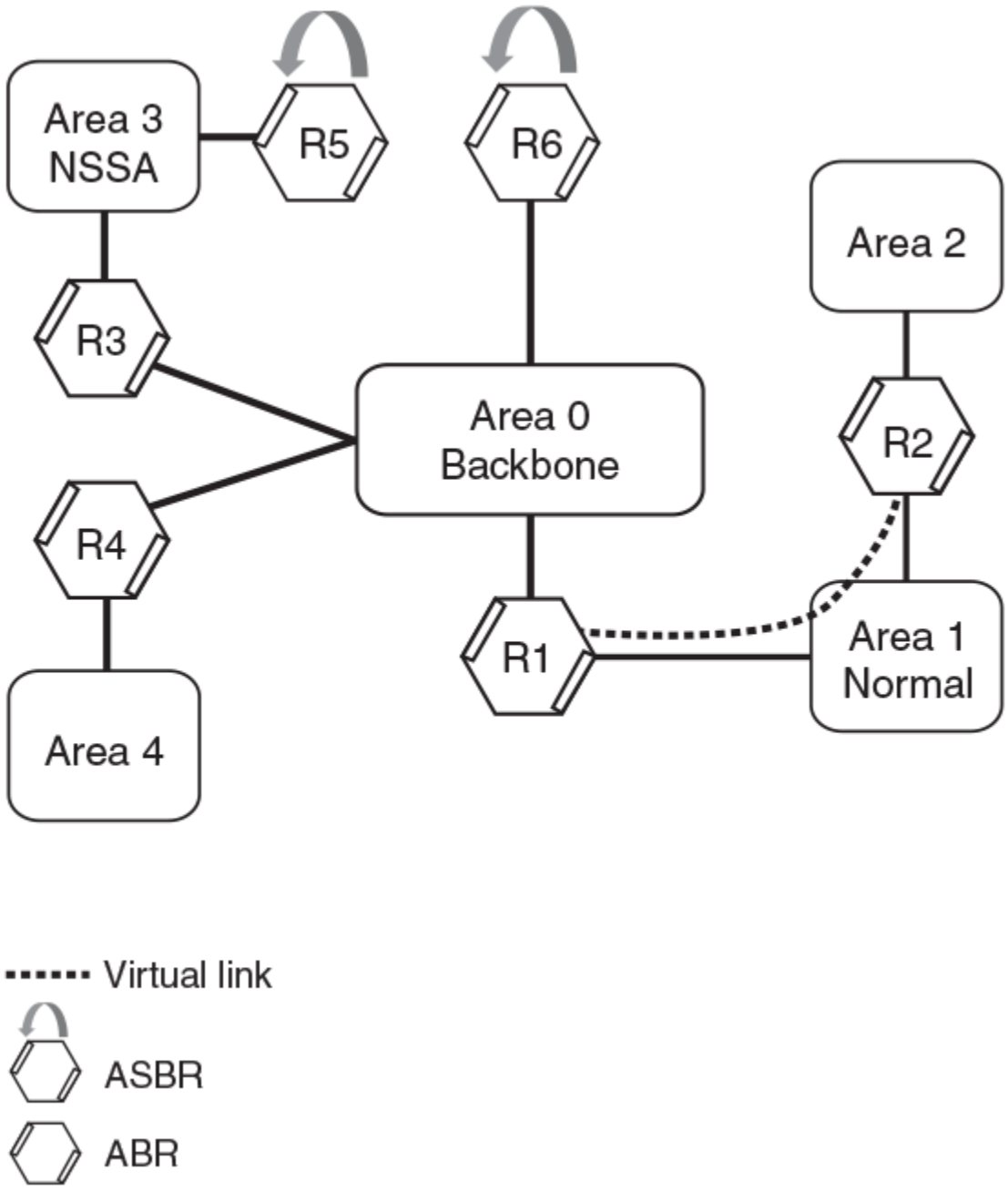
Virtual links

All ABRs must have either a direct or indirect link to an OSPFv3 backbone area (0 or 0.0.0.0). 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.

In the following figure, a virtual link has been created between ABR1 and ABR2. ABR1 has a direct link to the backbone area, while ABR2 has an indirect link to the backbone area through Area 1.

FIGURE 19 OSPFv3 virtual link



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.
- The neighbor router is the router ID of the device that is physically connected to the backbone when assigned from the router interface requiring a logical connection. The neighbor router is the router ID (IPv4 address) of the router requiring a logical connection to the backbone when assigned from the router interface with the physical connection.

When you establish an area virtual link, you must configure it on both ends of the virtual link. For example, imagine that ABR1 in Area 1 and Area 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.

Virtual links cannot be configured in stub areas and NSSAs.

Virtual link source address assignment

When devices at both ends of a virtual link communicate with one another, a global IPv6 address is automatically selected for each end device and this address is advertised into the transit area as an intra-area-prefix LSA.

The automatically selected global IPv6 address for that router is the first global address of any loopback interface in that transit area. If no global IPv6 address is available on a loopback interface in the area, the first global IPv6 address of the lowest-numbered interface in the UP state (belonging to the transit area) is assigned. If no global IPv6 address is configured on any of the OSPFv3 interfaces in the transit area, the virtual links in the transit area do not operate. The automatically selected IPv6 global address is updated whenever the previously selected IPv6 address of the interface changes, is removed, or if the interface goes down.

> NOTE
> The existing selected virtual link address does not change because the global IPv6 address is now available on a loopback interface or a lower-numbered interface in the transit area. To force the global IPv6 address for the virtual link to be the global IPv6 address of a newly configured loopback, or a lower-numbered interface in the area, you must either disable the existing selected interface or remove the currently selected global IPv6 address from the interface.

Configuring virtual links

If an Area Border Router (ABR) does not have a physical link to a backbone area, a virtual link can be configured between that ABR and another device within the same area that has a physical link to a backbone area.

A virtual link is configured, and a virtual link endpoint on two devices, ABR1 and ABR2, is defined.

1. On ABR1, enter the **configure terminal** command to access global configuration mode.

```
device# configure terminal
```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

```
device(config)# rbridge-id 122
```

3. Enter the **ip router-id** command to specify the router ID.

```
device(config-rbridge-id-122)# ip router-id 10.1.1.1
```

4. Enter the **ipv6 router ospf** command to enter OSPFv3 router configuration mode and enable OSPFv3 on the device.

```
device(config-rbridge-id-122)# ipv6 router ospf
```

5. Enter the **area** command to assign an OSPFv3 area ID.

```
device(config-ipv6-router-ospf-vrf-default-vrf)# area 0
```

6. Enter the **area** command to assign an OSPFv3 area ID.

```
device(config-ipv6-router-ospf-vrf-default-vrf)# area 1
```

7. Enter the **area virtual-link** command and the ID of the OSPFv3 device at the remote end of the virtual link to configure the virtual link endpoint.

```
device(config-ipv6-router-ospf-vrf-default-vrf)# area 1 virtual-link 10.2.2.2
```

8. On ABR2, enter the **configure terminal** command to access global configuration mode.

```
device# configure terminal
```

9. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

```
device(config)# rbridge-id 104
```

10. Enter the **ip router-id** command to specify the router ID.

```
device(config-rbridge-id-104)# ip router-id 10.2.2.2
```

11. Enter the **ipv6 router ospf** command to enter OSPFv3 router configuration mode and enable OSPFv3 on the device.

```
device(config-rbridge-id-104)# ipv6 router ospf
```

12. Enter the **area** command to assign an OSPFv3 area ID.

```
device(config-ipv6-router-ospf-vrf-default-vrf)# area 1
```

13. Enter the **area** command to assign an OSPFv3 area ID.

```
device(config-ipv6-router-ospf-vrf-default-vrf)# area 2
```

14. Enter the **area virtual-link** command and the ID of the OSPFv3 device at the remote end of the virtual link to configure the virtual link endpoint.

```
device(config-ipv6-router-ospf-vrf-default-vrf)# area 1 virtual-link 10.1.1.1
```

The following example configures a virtual link between two devices.

```
ABR1:
device1# configure terminal
device1(config)# rbridge-id 122
device1(config-rbridge-id-122)# ip router-id 10.1.1.1
device1(config-rbridge-id-122)# ipv6 router ospf
device1(config-ipv6-router-ospf-vrf-default-vrf)# area 0
device1(config-ipv6-router-ospf-vrf-default-vrf)# area 1
device1(config-ipv6-router-ospf-vrf-default-vrf)# area 1 virtual-link 10.2.2.2

ABR2:
device2# configure terminal
device2(config)# rbridge-id 104
device2(config-rbridge-id-104)# ip router-id 10.2.2.2
device2(config-rbridge-id-104)# ipv6 router ospf
device2(config-ipv6-router-ospf-vrf-default-vrf)# area 1
device2(config-ipv6-router-ospf-vrf-default-vrf)# area 2
device2(config-ipv6-router-ospf-vrf-default-vrf)# area 1 virtual-link 10.1.1.1
```

OSPFv3 route redistribution

Routes from various sources can be redistributed into OSPFv3. These routes can be redistributed in a number of ways.

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

- IPv6 static routes

- Directly connected IPv6 networks
- BGP4+

You can redistribute routes in the following ways:

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

  NOTE
  You must configure the route map before you configure a redistribution filter that uses 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 the **set metric** command inside the route map or the **default-metric** command. For a route redistributed without using a route map, the metric is set by the metric parameter if set or the **default-metric** command if the metric parameter is not set.

Redistributing routes into OSPFv3

OSPFv3 routes can be redistributed, and the routes to be redistributed can be specified.

The redistribution of both static routes and BGP routes into OSPFv3 is configured on device1. The redistribution of connected routes into OSPFv3 is configured on device2, and the connected routes to be redistributed are specified.

1. On device1, enter the **configure terminal** command to access global configuration mode.

   ```
   device# configure terminal
   ```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

   ```
   device(config)# rbridge-id 122
   ```

3. Enter the **ipv6 router ospf** command to enter OSPFv3 router configuration mode and enable OSPFv3 on the device.

   ```
   device(config-rbridge-id-122)# ipv6 router ospf
   ```

4. Enter the **redistribute** command with the **static** parameter to redistribute static routes.

   ```
   device(config-ipv6-router-ospf-vrf-default-vrf)# redistribute static
   ```

5. Enter the **redistribute** command with the **bgp** parameter to redistribute static routes.

   ```
   device(config-ipv6-router-ospf-vrf-default-vrf)# redistribute bgp
   ```

6. On device2, enter the **configure terminal** command to access global configuration mode.

   ```
   device# configure terminal
   ```

7. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

   ```
   device(config)# rbridge-id 122
   ```

8. Enter the **ipv6 router ospf** command to enter OSPFv3 router configuration mode and enable OSPFv3 on the device.

   ```
   device(config-rbridge-id-122)# ipv6 router ospf
   ```

9. Enter the **redistribute** command with the **connected** and **route-map** parameters to redistribute connected routes and specify a route map.

```
device(config-ipv6-router-ospf-vrf-default-vrf)# redistribute connected route-map rmap1
```

The following example redistributes static and BGP routes routes into OSPFv3 on a device.

```
device# configure terminal
device(config)# rbridge-id 122
device(config-rbridge-id-122)# ipv6 router ospf
device(config-ipv6-router-ospf-vrf-default-vrf)# redistribute static
device(config-ipv6-router-ospf-vrf-default-vrf)# redistribute bgp
```

The following example redistributes connected routes into OSPFv3 on a device and specifies a route map.

```
device# configure terminal
device(config)# rbridge-id 122
device(config-rbridge-id-122)# ipv6 router ospf
device(config-ipv6-router-ospf-vrf-default-vrf)# redistribute connected route-map rmap1
```

Default route origination

When the device is an OSPFv3 Autonomous System Boundary Router (ASBR), you can configure it to automatically generate a default external route into an OSPFv3 routing domain.

By default, a device does not advertise the default route into the OSPFv3 domain. If you want the device to advertise the OSPFv3 default route, you must explicitly enable default route origination. When you enable OSPFv3 default route origination, the device advertises a type 5 default route that is flooded throughout the autonomous system, with the exception of stub areas.

The device advertises the default route into OSPFv3 even if OSPFv3 route redistribution is not enabled, and even if the default route is learned through an IBGP neighbor. The device does not, however, originate the default route if the active default route is learned from an OSPFv3 router in the same domain.

> NOTE
> The device does not advertise the OSPFv3 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 OSPFv3 devices are not affected. If you re-enable the default route origination, the change takes effect immediately and you do not need to reload the software.

Configuring default external routes

OSPFv3 default routes can be created and advertised.

1. Enter the **configure terminal** command to access global configuration mode.

```
device# configure terminal
```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

```
device(config)# rbridge-id 122
```

3. Enter the **ipv6 router ospf** command to enter OSPFv3 router configuration mode and enable OSPFv3 globally.

```
device(config-rbridge-id-122)# ipv6 router ospf
```

4. Enter the **default-information-originate** command with the **always**, **metric**, and **metric-type** parameters.

```
device(config-ipv6-router-ospf-vrf-default-vrf)# default-information-originate always metric 2
metric-type type1
```

A default type 1 external route with a metric of 2 is created and advertised.

The following example creates and advertises a default route with a metric of 2 and a type 1 external route.

```
device# configure terminal
device(config)# rbridge-id 122
device(config-rbridge-id-122)# ipv6 router ospf
device(config-ipv6-router-ospf-vrf-default-vrf)# default-information-originate always metric 2 metric-type
type1
```

Disabling and re-enabling OSPFv3 event logging

OSPFv3 event logging, such as neighbor state changes and database overflow conditions, can be disabled and re-enabled.

1. Enter the **configure terminal** command to access global configuration mode.

```
device# configure terminal
```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

```
device(config)# rbridge-id 122
```

3. Enter the **ipv6 router ospf** command to enter OSPFv3 configuration mode and enable OSPFv3 globally.

```
device(config-rbridge-id-122)# ipv6 router ospf
```

4. Enter the **no log** command to disable the logging of OSPFv3 events.

```
device(config-ipv6-router-ospf-vrf-default-vrf)# no log
```

The following example re-enables the logging of OSPFv3 events.

```
device# configure terminal
device(config)# rbridge-id 122
device(config-rbridge-id-122)# ipv6 router ospf
device(config-ipv6-router-ospf-vrf-default-vrf)# log all
```

Filtering OSPFv3 routes

You can filter the routes to be placed in the OSPFv3 route table by configuring distribution lists.

The functionality of OSPFv3 distribution lists is similar to that of OSPFv2 distribution lists. Refer to the **OSPFv2** chapter for more information.

SPF timers

The device uses an SPF delay timer and an SPF hold-time timer to calculate the shortest path for OSPFv3 routes. The values for both timers can be changed.

The device uses the following timers when calculating the shortest path for OSPFv3 routes:

- SPF delay: When the device receives a topology change, it waits before starting a Shortest Path First (SPF) calculation. By default, the device waits 5 seconds. You can configure the SPF delay to a value from 0 through 65535 seconds. If you set the SPF delay to 0 seconds, the device immediately begins the SPF calculation after receiving a topology change.
- SPF hold time: The device waits a specific amount of time between consecutive SPF calculations. By default, it waits 10 seconds. You can configure the SPF hold time to a value from 0 through 65535 seconds. If you set the SPF hold time to 0 seconds, the device 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 device does not accept only one timer value.

Modifying SPF timers

The Shortest Path First (SPF) delay and hold time can be modified.

1. Enter the **configure terminal** command to access global configuration mode.

```
device# configure terminal
```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode..

```
device(config)# rbridge-id 122
```

3. Enter the **ipv6 router ospf** command to enter OSPFv3 router configuration mode and enable OSPFv3 globally.

```
device(config-rbridge-id-122)# ipv6 router ospf
```

4. Enter the **timers** command with the **spf** parameter.

```
device(config-ipv6-router-ospf-vrf-default-vrf)# timers spf 1 5
```

The SPF delay is changed to 1 second and the SPF hold time is changed to 5 seconds.

The following example changes the SPF delay and hold time.

```
device# configure terminal
device(config)# rbridge-id 122
device(config-rbridge-id-122)# ipv6 router ospf
device(config-ipv6-router-ospf-vrf-default-vrf)# timers spf 1 5
```

OSPFv3 administrative distance

Devices can learn about networks from various protocols and select a route based on the source of the route information. This decision can be influenced if the default administrative distance for OSPFv3 routes is changed. Consequently, the routes to a network may differ depending on the protocol from which the routes were learned.

You can influence the device's decision by changing the default administrative distance for OSPFv3 routes. You can configure a unique administrative distance for each type of OSPFv3 route. For example, you can configure the Extreme device to prefer a static route over an OSPFv3 inter-area route and to prefer OSPFv3 intra-area routes over 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

NOTE
The choice of routes within OSPFv3 is not influenced. For example, an OSPFv3 intra-area route is always preferred over an OSPFv3 inter-area route, even if the intra-area route's distance is greater than the inter-area route's distance.

Configuring administrative distance based on route type

The default administrative distances for intra-area routes, inter-area routes, and external routes can be altered.

1. Enter the **configure terminal** command to access global configuration mode.

```
device# configure terminal
```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

```
device(config)# rbridge-id 122
```

3. Enter the **ipv6 router ospf** command to enter OSPFv3 router configuration mode and enable OSPFv3 globally.

```
device(config-rbridge-id-122)# ipv6 router ospf
```

4. Enter the **distance** command with the **intra-area** parameter.

```
device(config-ipv6-router-ospf-vrf-default-vrf)# distance intra-area 80
```

The administrative distance for intra-area routes is changed from the default to 80.

5. Enter the **distance** command with the **inter-area** parameter.

```
device(config-ipv6-router-ospf-vrf-default-vrf)# distance inter-area 90
```

The administrative distance for inter-area routes is changed from the default to 90.

6. Enter the **distance** command with the **external** parameter.

```
device(config-ipv6-router-ospf-vrf-default-vrf)# distance external 100
```

The administrative distance for external routes is changed from the default to 100.

The following example changes the default administrative distances for intra-area routes, inter-area routes, and external routes.

```
device# configure terminal
device(config)# rbridge-id 122
device(config-rbridge-id-122)# ipv6 router ospf
device(config-ipv6-router-ospf-vrf-default-vrf)# distance intra-area 80
device(config-ipv6-router-ospf-vrf-default-vrf)# distance inter-area 90
device(config-ipv6-router-ospf-vrf-default-vrf)# distance external 100
```

Changing the reference bandwidth for the cost on OSPFv3 interfaces

The reference bandwidth for OSPFv3 can be altered, resulting in various costs.

1. Enter the **configure terminal** command to access global configuration mode.

   ```
   device# configure terminal
   ```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

   ```
   device(config)# rbridge-id 122
   ```

3. Enter the **ipv6 router ospf** command to enter OSPFv3 router configuration mode and enable OSPFv3 globally.

   ```
   device(config-rbridge-id-122)# ipv6 router ospf
   ```

4. Enter the **auto-cost reference-bandwidth** command to change the reference bandwidth.

   ```
   device(config-ipv6-router-ospf-vrf-default-vrf)# auto-cost reference-bandwidth 500
   ```

The following example changes the auto-cost reference bandwidth to 500.

```
device# configure terminal
device(config)# rbridge-id 122
device(config-rbridge-id-122)# ipv6 router ospf
device(config-ipv6-router-ospf-vrf-default-vrf)# auto-cost reference-bandwidth 500
```

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

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

OSPFv3 LSA refreshes

To prevent a refresh from being performed each time an individual LSA's refresh timer expires, OSPFv3 LSA refreshes are delayed for a specified time interval. This pacing interval can be altered.

The 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 device refreshes and sends out together in one or more packets.

The pacing interval, which is the interval at which the device refreshes an accumulated group of LSAs, is configurable in a range from 10 through 1800 seconds (30 minutes). The default is 240 seconds (4 minutes). Thus, every four minutes, the device refreshes the group of accumulated LSAs and sends the group together in the same packets.

The pacing interval is inversely proportional to the number of LSAs the 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 to 100 LSAs), increasing the pacing interval to 10 to 20 minutes may enhance performance only slightly.

Configuring the OSPFv3 LSA pacing interval

The interval between OSPFv3 LSA refreshes can be modified.

1. Enter the **configure terminal** command to access global configuration mode.

   ```
   device# configure terminal
   ```

2. Enter the **rbridge-id** command with an RBridge ID t o enter RBridge ID configuration mode.

   ```
   device(config)# rbridge-id 122
   ```

3. Enter the **ipv6 router ospf** command to enter OSPFv3 router configuration mode and enable OSPFv3 globally.

   ```
   device(config-rbridge-id-122)# ipv6 router ospf
   ```

4. Enter the **timers** command with the **lsa-group-pacing** parameter.

   ```
   device(config-ipv6-router-ospf-vrf-default-vrf)# timers lsa-group-pacing 120
   ```

   The OSPFv3 LSA pacing interval is changed to 120 seconds (two minutes).

The following example restores the pacing interval to the default value of 240 seconds (4 minutes).

```
device# configure terminal
device(config)# rbridge-id 122
device(config-rbridge-id-122)# ipv6 router ospf
device(config-ipv6-router-ospf-vrf-default-vrf)# no timers lsa-group-pacing
```

External route summarization

An ASBR can be configured to advertise one external route as an aggregate for all redistributed routes that are covered by a specified IPv6 summary address range.

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

If a route that falls within a configured summary address range is imported by the 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 summary address range is removed by the device, no action is taken if there are other imported routes that fall within the same summary address range; otherwise, the aggregate route is flushed.

You can configure up to 4 summary address ranges.

The device sets the forwarding address of the aggregate route to 0 and sets the tag to 0. If you delete a summary 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 (LSDB) overflow condition occurs, all aggregate routes and other external routes are flushed out of the autonomous system. When the 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 summary address ranges, the device applies the redistribution filters to routes first, and then applies them to the summary address ranges.

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

NOTE
Only imported, type 5 external LSA routes are affected. A single type 5 LSA is generated and flooded throughout the autonomous system for multiple external routes.

OSPFv3 over VRF

OSPFv3 can run over multiple Virtual Routing and Forwarding (VRF) instances. OSPFv3 maintains multiple instances of the routing protocol to exchange route information among various VRF instances. A multi-VRF-capable router maps an input interface to a unique VRF, based on user configuration. These input interfaces can be physical or a virtual interface. By default, all input interfaces are attached to the default VRF instance. All OSPFv3 commands are available over default and nondefault VRF instances.

Multi-VRF for OSPF (also known as VRF-Lite for OSPF) provides a reliable mechanism for trusted VPNs to be built over a shared infrastructure. The ability to maintain multiple virtual routing or forwarding tables allows overlapping private IP addresses to be maintained across VPNs.

Enabling OSPFv3 in a non-default VRF

When OSPFv3 is enabled in a non-default VRF instance, the device enters OSPFv3 router VRF configuration mode. Several commands can then be accessed that allow the configuration of OSPFv3.

1. Enter the **configure terminal** command to access global configuration mode.

```
device# configure terminal
```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

```
device(config)# rbridge-id 122
```

3. Enter the **vrf** command and specify a name to enter Virtual Routing and Forwarding (VRF) configuration mode and create a non-default VRF instance.

```
device(config-rbridge-id-122)# vrf green
```

4. Enter the **ip router-id** command to specify the router ID.

```
device(config-vrf-green)# ip router-id 10.11.12.14
```

5. Enter the **address-family ipv6 unicast** command to enter IPv6 address-family configuration mode.

```
device(config-vrf-green)# address-family ipv6 unicast
```

6. Enter the **exit** command until you return to RBridge ID configuration mode.

```
device(vrf-ipv6-unicast)# exit
```

7. Enter the **ipv6 router ospf** command and specify a VRF name to enter OSPFv3 router VRF configuration mode and enable OSPFv3 on a non-default VRF.

```
device(config-rbridge-id-122)# ipv6 router ospf vrf green
```

The following example enables OSPFv3 in a non-default VRF.

```
device# configure terminal
device(config)# rbridge-id 122
device(config-rbridge-id-122)# vrf green
device(config-vrf-green)# ip router-id 10.11.12.14
device(config-vrf-green)# address-family ipv6 unicast
device(vrf-ipv6-unicast)#
device(vrf-ipv6-unicast)# exit
device(config-vrf-green)# exit
device(config-rbridge-id-122)# ipv6 router ospf vrf green
device(config-ipv6-router-ospf-vrf-green)#
```

Assigning OSPFv3 areas in a non-default VRF

Areas can be assigned as OSPFv3 areas in a non-default VRF.

Enable IPv6 on each interface on which you plan to enable OSPFv3. You enable IPv6 on an interface by configuring an IP address or explicitly enabling IPv6 on that interface.

1. Enter the **configure terminal** command to access global configuration mode.

```
device# configure terminal
```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

```
device(config)# rbridge-id 122
```

3. Enter the **vrf** command and specify a name to enter Virtual Routing and Forwarding (VRF) configuration mode and create a non-default VRF instance.

```
device(config-rbridge-id-122)# vrf red
```

4. Enter the **ip router-id** command to specify the router ID.

```
device(config-vrf-red)# ip router-id 10.11.12.14
```

5. Enter the **address-family ipv6 unicast** command to enter IPv6 address-family configuration mode.

```
device(config-vrf-red)# address-family ipv6 unicast
```

6. Enter the **exit** command until you return to RBridge ID configuration mode.

```
device(vrf-ipv6-unicast)# exit
```

7. Enter the **ipv6 router ospf** command and specify a VRF name to enter OSPFv3 configuration mode and enable OSPFv3 in a non-default VRF.

```
device(config-rbridge-id-122)# ipv6 router ospf vrf red
```

8. Enter the **area** command to define an OSPFv3 area ID.

```
device(config-ipv6-router-ospf-vrf-red)# area 0
```

9. Enter the **area** command to define a second OSPFv3 area ID.

```
device(config-ipv6-router-ospf-vrf-red)# area 10.1.1.1
```

The following example assigns an OSPFv3 ID to two areas in a non-default VRF instance. One of the areas is assigned by decimal number. The second area is assigned by IP address.

```
device# configure terminal
device(config)# rbridge-id 122
device(config-rbridge-id-122)# vrf red
device(config-vrf-red)# ip router-id 10.11.12.13
device(config-vrf-red)# address-family ipv6 unicast
device(vrf-ipv6-unicast)#
device(vrf-ipv6-unicast)# exit
device(config-vrf-red)# exit
device(config-rbridge-id-122)# ipv6 router ospf vrf red
device(config-ipv6-router-ospf-vrf-red)# area 0
device(config-ipv6-router-ospf-vrf-red)# area 10.1.1.1
```

Setting all OSPFv3 interfaces to the passive state

All OSPFv3 interfaces can be set as passive, causing them to drop all OSPFv3 control packets.

1. Enter the **configure terminal** command to access global configuration mode.

```
device# configure terminal
```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

```
device(config)# rbridge-id 122
```

3. Enter the **ipv6 router ospf** command to enter OSPFv3 router configuration mode and enable OSPFv3 globally.

```
device(config-rbridge-id-122)# ipv6 router ospf
```

4. Enter the **default-passive-interface** command to mark all interfaces passive by default.

```
device(config-ipv6-router-ospf-vrf-default-vrf)# default-passive-interface
```

The following example sets all OSPFv3 interfaces for a specified RBridge as passive, causing them to drop all the OSPFv3 control packets.

```
device# configure terminal
device(config)# rbridge-id 122
device(config-rbridge-id-122)# ipv6 router ospf
device(config-ipv6-router-ospf-vrf-default-vrf)# default-passive-interface
```

OSPFv3 graceful restart helper

The OSPFv3 graceful restart (GR) helper provides a device with the capability to participate in a graceful restart in helper mode so that it assists a neighboring routing device that is performing a graceful restart.

When OSPFv3 GR helper is enabled on a device, the device enters helper mode upon receipt of a grace-LSA where the neighbor state is full. By default, the helper capability is enabled when you start OSPFv3, even if graceful restart is not supported.

Disabling OSPFv3 graceful restart helper

The OSPFv3 graceful restart (GR) helper is enabled by default, and can be disabled on a routing device.

1. Enter the **configure terminal** command to access global configuration mode.

   ```
   device# configure terminal
   ```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

   ```
   device(config)# rbridge-id 122
   ```

3. Enter the **ipv6 router ospf** command to enter OSPFv3 router configuration mode and enable OSPFv3 globally.

   ```
   device(config-rbridge-id-122)# ipv6 router ospf
   ```

4. Enter the **no graceful-restart helper** command with the **strict-lsa-checking** to disable the GR helper with strict link-state advertisement (LSA) checking.

   ```
   device(config-ipv6-router-ospf-vrf-default-vrf)# no graceful-restart helper strict-lsa-checking
   ```

The following example disables the GR helper with strict link-state advertisement (LSA) checking.

```
device# configure terminal
device(config)# rbridge-id 122
device(config-rbridge-id-122)# ipv6 router ospf
device(config-ipv6-router-ospf-vrf-default-vrf)# no graceful-restart helper strict-lsa-checking
```

Re-enabling OSPFv3 graceful restart helper

If the OSPFv3 graceful restart (GR) helper has been disabled on a routing device, it can be re-enabled. GR helper mode can also be enabled with strict link-state advertisement (LSA) checking.

1. Enter the **configure terminal** command to access global configuration mode.

   ```
   device# configure terminal
   ```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

   ```
   device(config)# rbridge-id 122
   ```

3. Enter the **ipv6 router ospf** command to enter OSPFv3 router configuration mode and enable OSPFv3 globally.

```
device(config-rbridge-id-122)# ipv6 router ospf
```

4. Enter the **graceful-restart helper** command and specify the **strict-lsa-checking** parameter to re-enable the GR helper with strict LSA checking.

```
device(config-ipv6-router-ospf-vrf-default-vrf)# graceful-restart helper strict-lsa-checking
```

The following example re-enables the GR helper with strict LSA checking.

```
device# configure terminal
device(config)# rbridge-id 122
device(config-rbridge-id-122)# ipv6 router ospf
device(config-ipv6-router-ospf-vrf-default-vrf)# graceful-restart helper strict-lsa-checking
```

OSPFv3 non-stop routing

OSPFv3 can continue operation without interruption during hitless failover when the NSR feature is enabled.

During graceful restart (GR), the restarting neighbors must help build routing information during a failover. However, the GR helper may not be supported by all devices in a network. Non-stop routing (NSR) eliminates this dependency.

NSR does not require support from neighboring devices to perform hitless failover, and OSPF can continue operation without interruption.

> NOTE
> NSR does not support IPv6-over-IPv4 tunnels and virtual links, so traffic loss is expected while performing hitless failover.

Enabling OSPFv3 NSR

OSPFv3 non-stop routing (NSR) can be re-enabled if it has been disabled. The following task re-enables NSR for OSPFv3.

1. Enter the **configure terminal** command to access global configuration mode.

```
device# configure terminal
```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

```
device(config)# rbridge-id 122
```

3. Enter the **ipv6 router ospf** command to enter OSPFv3 router configuration mode and enable OSPFv3 globally.

```
device(config-rbridge-id-122)# ipv6 router ospf
```

4. Enter the **graceful restart** command to re-enable GR on the device.

```
device(config-ipv6-router-ospf-vrf-default-vrf)# nonstop-routing
```

The following example re-enables NSR for OSPFv3.

```
device# configure terminal
device(config)# rbridge-id 122
device(config-rbridge-id-122)# ipv6 router ospf
device(config-ipv6-router-ospf-vrf-default-vrf)# nonstop-routing
```

OSPFv3 max-metric router LSA

OSPFv3 can be configured to advertise its locally generated router LSAs with a maximum metric to direct transit traffic away from the device, while still routing for directly connected networks.

By advertising the maximum metric, the device does not attract transit traffic. A device which does not handle transit traffic, and only forwards packets destined for its directly connected links, is known as a stub router. In OSPFv3 networks, a device could be placed in a stub router role by advertising large metrics for its connected links, so that the cost of a path through the device becomes larger than that of an alternative path.

You can configure OSPFv3 max-metric router LSA in either startup or non-startup mode. Configuring max-metric on startup may be helpful on ASBRs where protocols such as BGP converge after OSPF converges. Configuring max-metric on non-startup may be helpful in database overflow scenarios.

> NOTE
> The **on-startup** configuration does not apply to NSR restarts.

Configuring the OSPFv3 max-metric router LSA

By configuring the OSPFv3 max-metric router LSA feature you can enable OSPFv3 to advertise its locally generated router LSAs with a maximum metric.

> NOTE
> You can configure OSPFv3 max-metric router LSA in either startup or non-startup mode. Configuring max-metric with non-startup mode, it is only applied once and is not persistent across reloads, or after the **clear ip ospf all** command is issued.

1. Enter the **configure terminal** command to access global configuration mode.

```
device# configure terminal
```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

```
device(config)# rbridge-id 122
```

3. Enter the **ip router-id** command to specify the router ID.

```
device(config-rbridge-id-122)# ip router-id 10.11.12.13
```

4. Enter the **ipv6 router ospf** command to enter OSPFv3 router configuration mode and enable OSPFv3 on the device.

```
device(config-rbridge-id-122)# ipv6 router ospf
```

5. Enter the **max-metric router-lsa** command with the **external-lsa** keyword and specify a value to configure the maximum metric value .

```
device(config-ipv6-router-ospf-vrf-default-vrf)# max-metric router-lsa external-lsa 1500
```

6. Enter the **max-metric router-lsa** command with the **on-startup** keyword and specify a value to specify a period of time to advertise a maximum metric after a restart before advertising with a normal metric.

```
device(config-ipv6-router-ospf-vrf-default-vrf)# max-metric router-lsa on-startup 85
```

The following example configures an OSPFv3 device to advertise a maximum metric and sets the maximum metric value for external LSAs to 1500, and configures the device to advertise a maximum metric for 85 seconds after a restart before advertising with a normal metric.

```
device# configure terminal
device(config)# rbridge-id 122
device(config-rbridge-id-122)# ip router-id 10.11.12.13
device(config-rbridge-id-122)# ipv6 router ospf
device(config-ipv6-router-ospf-vrf-default-vrf)# max-metric router-lsa external-lsa 1500
device(config-ipv6-router-ospf-vrf-default-vrf)# max-metric router-lsa on-startup 85
```

IPsec for OSPFv3

IP Security (IPsec) secures OSPFv3 communications by authenticating and encrypting each IP packet of a communication session.

IPsec provides security features such as authentication of data origin, data integrity, replay protection, and message confidentiality. You can use IPsec to secure specific OSPFv3 areas and interfaces and protect OSPFv3 virtual links.

IPsec for OSPFv3 constitutes two basic protocols to authenticate routing information between peers:

- **Authentication header (AH)**: AH provides data origin authentication and connectionless integrity, as well as providing the optional replay protection feature. AH authenticates as much of the IP header as possible, as well as the upper-level protocol data such as source IPv6 address, destination IPv6 address, flags, and IP payload. However, some IP header fields, such as TTL and checksum are often modified in transit and, therefore, cannot be protected by AH.
- **Encapsulating Security Payload (ESP)**: ESP can provide message confidentiality, connectionless data integrity, and optional replay protection. ESP has both a header and a trailer. The authentication data of ESP cannot protect the outer IP header, only the payload that is being encrypted.

IPsec is available for OSPFv3 traffic only and only for packets that are “for-us”. A for-us packet is addressed to one of the IPv6 addresses on the device or to an IPv6 multicast address. Packets that are only forwarded by the line card do not receive IPsec scrutiny.

The devices support the following components of IPsec for IPv6-addressed packets:

- Authentication through AH
- Authentication through ESP in transport mode
- Hashed Message Authentication Code-Secure Hash Algorithm 1 (HMAC-SHA-1) as the authentication algorithm
- Hashed Message Authentication Code-Message Digest 5 (HMAC-MD5) as the authentication algorithm
- Security parameter index (SPI)
- Manual configuration of keys
- Configurable rollover timer

IPsec can be enabled on the following logical entities:

- Interface
- Area
- Virtual link

IPsec is based on security associations (SAs). With respect to traffic classes, this implementation of IPsec uses a single security association between the source and destination to support all traffic classes and does not differentiate between the different classes of traffic that the DSCP bits define.

IPsec on a virtual link is a global configuration. Interface and area IPsec configurations are more granular.

Among the entities that can have IPsec protection, the interfaces and areas can overlap. The interface IPsec configuration takes precedence over the area IPsec configuration when an area and an interface within that area use IPsec. Therefore, if you configure IPsec

for an interface and an area configuration also exists that includes this interface, the interface's IPsec configuration is used by that interface. However, if you disable IPsec on an interface, IPsec is disabled on the interface even if the interface has its own specific authentication.

For IPsec, the system generates two types of databases. The Security Association Database (SAD) contains a security association for each interface or one global database for a virtual link. Even if IPsec is configured for an area, each interface that uses the area's IPsec still has its own security association in the SAD. Each SA in the SAD is a generated entry that is based on your specifications of an authentication protocol, destination address, and a security parameter index (SPI). The SPI number is user-specified according to the network plan. Consideration for the SPI values to specify must apply to the whole network.

The system-generated security policy databases (SPDs) contain the security policies against which the system checks the for-us packets. For each for-us packet that has an ESP header, the applicable security policy in the security policy database (SPD) is checked to see if this packet complies with the policy. The IPsec task drops the non-compliant packets. Compliant packets continue on to the OSPFv3 task.

IPsec for OSPFv3 configuration

IPsec authentication can be enabled on both default and nondefault VRFs. IPsec authentication is disabled by default.

The following IPsec parameters are configurable:

- AH security protocol
- ESP protocol
- Authentication
- Hashed Message Authentication Code-Message Digest 5 (HMAC-MD5) authentication algorithm
- Hashed Message Authentication Code-Secure Hash Algorithm 1 (HMAC-SHA-1) authentication algorithm
- Security parameter index (SPI)
- A 40-character key using hexadecimal characters
- An option for not encrypting the keyword when it appears in **show** command output
- Key rollover timer
- Specifying the key add remove timer

Configuring IPsec on an OSPFv3 area

IPsec can be configured to secure communications on an OSPFv3 area.

NOTE
When IPsec is configured for an area, the security policy is applied to all the interfaces in the area.

1. Enter the **configure terminal** command to access global configuration mode.

```
device# configure terminal
```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

```
device(config)# rbridge-id 122
```

3. Enter the **ip router-id** command to specify the router ID.

```
device(config-rbridge-id-122)# ip router-id 10.11.12.13
```

4. Enter the **ipv6 router ospf** command to enter OSPFv3 configuration mode and enable OSPFv3 on the device.

```
device(config-rbridge-id-122)# ipv6 router ospf
```

5. Enter **area authentication spi** *spi* **ah hmac-md5 key**, specifying an area, and enter a 40-character hexadecimal key.

```
device(config-ipv6-router-ospf-vrf-default-vrf)# area 0 authentication spi 600 ah hmac-md5 key
abcef12345678901234fedcba098765432109876
```

IPsec is configured in OSPv3 area 0 with a security parameter index (SPI) value of 600, and the authentication header (AH) protocol is selected. Message Digest 5 (MD5) authentication on the area is enabled.

The following example enables AH and MD5 authentication for the OSPFv3 area, setting an SPI value of 600.

```
device# configure terminal
device(config)# rbridge-id 122
device(config-rbridge-id-122)# ip router-id 10.11.12.13
device(config-rbridge-id-122)# ipv6 router ospf
device(config-ipv6-router-ospf-vrf-default-vrf)# area 0 authentication spi 600 ah hmac-md5 key
abcef12345678901234fedcba098765432109876
```

Configuring IPsec on an OSPFv3 interface

IPsec can be configured to secure communications on an OSPFv3 interface.

For IPsec to work, the IPsec configuration must be the same on all the routers to which an interface connects.

NOTE

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

1. Enter the **configure terminal** command to access global configuration mode.

```
device# configure terminal
```

2. Enter the **interface** command and specify an interface.

```
device(config)# interface tengigabitethernet 1/0/1
```

3. Enter the **ipv6 ospf area** command to assign a specified area to the interface.

```
device(config-if-te-1/0/1)# ipv6 ospf area 0
```

4. Enter **ipv6 ospf authentication spi** *value* **esp null hmac-sha1** and specify a 40-character hexadecimal key.

```
device(config-if-te-1/0/1)# ipv6 ospf authentication spi 512 esp null hmac-sha1 key
abcef12345678901234fedcba098765432109876
```

IPsec is configured on the specified interface with a security parameter index (SPI) value of 512, and the Encapsulating Security Payload (ESP) protocol is selected. Secure Hash Algorithm 1 (SHA-1) authentication is enabled.

The following example enables ESP and SHA-1 on a specified OSPFv3 10-gigabit interface.

```
device# configure terminal
device(config)# interface tengigabitethernet 1/0/1
device(config-if-te-1/0/1)# ipv6 ospf area 0
device(config-if-te-1/0/1)# ipv6 ospf authentication spi 512 esp null hmac-sha1 key
abcef12345678901234fedcba098765432109876
```

Configuring IPsec on OSPFv3 virtual links

IP Security (IPsec) can be configured for virtual links.

An OSPFv3 virtual link must be configured.

The virtual link IPsec security associations (SAs) and policies are added to all interfaces of the transit area for the outbound direction. For the inbound direction, IPsec SAs and policies for virtual links are added to the global database.

1. Enter the **configure terminal** command to access global configuration mode.

```
device# configure terminal
```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

```
device(config)# rbridge-id 122
```

3. Enter the **ip router-id** command to specify the router ID.

```
device(config-rbridge-id-122)# ip router-id 10.1.1.1
```

4. Enter the **ipv6 router ospf** command to enter OSPFv3 configuration mode and enable OSPFv3 on the router.

```
device(config-rbridge-id-122)# ipv6 router ospf
```

5. Enter **area virtual-link authentication spi** *value* **ah hmac-sha1 no-encrypt** *key*, specifying an area address and the ID of the OSPFv3 device at the remote end of the virtual link.

```
device(config-ipv6-router-ospf-vrf-default-vrf)# area 1 virtual-link 10.1.1.1 authentication spi 512
ah hmac-sha1 no-encrypt key 1134567890223456789012345678901234567890
```

   IPsec is configured on the specified virtual link in OSPF area 1. The device ID associated with the virtual link neighbor is 10.1.1.1, the SPI value is 512, and the authentication header (AH) protocol is selected. Secure Hash Algorithm 1 (SHA-1) authentication is enabled. The 40-character key is not encrypted in **show** command displays.

The following example configures IPsec on an OSPFv3 area.

```
device# configure terminal
device(config)# rbridge-id 122
device(config-rbridge-id-122)# ip router-id 10.1.1.1
device(config-rbridge-id-122)# ipv6 router ospf
device(config-ipv6-router-ospf-vrf-default-vrf)# area 1 virtual-link 10.1.1.1 authentication spi 512 ah
hmac-sha1 no-encrypt key 1134567890223456789012345678901234567890
```

Specifying the key rollover and key add-remove timers

The key rollover timer can be configured so that rekeying takes place on all the nodes at the same time and the security parameters are consistent across all the nodes. The timing of the authentication key add-remove interval can also be altered.

1. Enter the **configure terminal** command to access global configuration mode.

```
device# configure terminal
```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

```
device(config)# rbridge-id 122
```

3. Enter the **ip router-id** command to specify the router ID.

```
device(config-rbridge-id-122)# ip router-id 10.11.12.13
```

4. Enter the **ipv6 router ospf** command to enter OSPFv3 router configuration mode and enable OSPFv3 on the device.

```
device(config-rbridge-id-122)# ipv6 router ospf
```

5. Enter the **key-add-remove-interval** command and specify the desired interval to set the timing of the authentication key add-remove interval.
6. Enter the **key-rollover-interval** command and specify the desired interval to set the timing of the configuration changeover.

```
device(config-ipv6-router-ospf-vrf-default-vrf)# key-rollover-interval 240
```

The following example sets the key add-remove interval to 240 seconds (4 minutes) and sets the timing of the configuration changeover to 240 seconds (4 minutes).

```
device# configure terminal
device(config)# rbridge-id 122
device(config-rbridge-id-122)# ip router-id 10.11.12.13
device(config-rbridge-id-122)# ipv6 router ospf
device(config-ipv6-router-ospf-vrf-default-vrf)# key-add-remove-interval 240
device(config-ipv6-router-ospf-vrf-default-vrf)# key-rollover-interval 240
```

Displaying OSPFv3 results

The **show ipv6 ospf** command and its variations can be used to display information about OSPFv3 configurations.

Use one or more of the following commands to verify OSPFv3 information. Using the **show ipv6 ospf** command is optional, and the variations of the command can be entered in any order.

1. Enter the **exit** command as necessary to access privileged EXEC mode.

```
device(config)# exit
```

2. Enter the **show ipv6 ospf** command to display general OSPFv3 information.

```
device# show ipv6 ospf

OSPFv3 Process number 0 with Router ID 0x01010101(1.1.1.1)
 Running 0 days 0 hours 24 minutes 36 seconds
 Number of AS scoped LSAs is 0
 Sum of AS scoped LSAs Checksum is 00000000
 External LSA Limit is 250000
 Database Overflow Interval is 10
 Database Overflow State is NOT OVERFLOWED
 Route calculation executed 34 times
 Pending outgoing LSA count 0
 Authentication key rollover interval 300 seconds
 Number of areas in this router is 2
 Router is operating as ABR
 High Priority Message Queue Full count: 0
 Graceful restart helper is enabled, strict lsa checking is disabled
 Nonstop Routing is disabled
```

The example output offers general OSPFv3 information and indicates that the device is not operating as an ASBR. If the device is not operating as an ASBR, there is no information about redistribution in the output.

3. The following example of the **show ipv6 ospf summary** command shows summary output for the one IPv6 OSPF session that is configured.

```
device# show ipv6 ospf summary

Total number of IPv6 OSPF instances: 1
Seq Instance        Intfs   Nbrs    Nbrs-Full LSAs    Routes
1   default-vrf     3       2       2         25      6
```

4. The following example of the **show ipv6 ospf area** command shows detailed output for assigned OSPFv3 Area 0.

```
device# show ipv6 ospf area 0

Area 0:
  Authentication: Not Configured
  Active interface(s)attached to this area: Te 1/0/1 VLink 1
  Inactive interface(s)attached to this area: None
  Number of Area scoped LSAs is 12
  Sum of Area LSAs Checksum is 0006e83c
  Statistics of Area 0:
    SPF algorithm executed 17 times
    SPF last updated: 333 sec ago
    Current SPF node count: 2
      Router: 2 Network: 0
      Maximum of Hop count to nodes: 1
```

5. The following example of the **show ipv6 ospf interface brief** command shows limited OSPFv3 interface information.

```
device# show ipv6 ospf interface brief
Interface     Area     Status Type Cost  State    Nbrs(F/C)
Te 1/0/1      0        up     BCST 1     DR       0/0
Te 1/0/2      1        up     BCST 1     BDR      1/1
```

6. The following example of the **show ipv6 ospf neighbor** command shows OSPFv3 neighbor information for the device.

```
device# show ipv6 ospf neighbor
Total number of neighbors in all states: 1
Number of neighbors in state Full      : 1
RouterID        Pri State    DR              BDR             Interface     [State]
2.2.2.2           1 Full     2.2.2.2         1.1.1.1         Te 1/0/2      [BDR]
```

7. The following example of the **show ipv6 ospf virtual-neighbor** command shows information about an OSPFv3 virtual neighbor.

```
device# show ipv6 ospf virtual-neighbor

Index Router ID        Address                       State     Interface
1     2.2.2.2          2001:2:0:2::2                 Full      Te 1/0/2
                Option: 00-00-00    QCount: 0    Timer: 46
```

8. The following example of the **show ipv6 ospf virtual-links** command shows information about OSPFv3 virtual links.

```
device# show ipv6 ospf virtual-links
Transit Area ID  Router ID        Interface Address             State
1                1.1.1.1          2001:2:0:2::2                 P2P
  Timer intervals(sec) :
    Hello 10, Hello Jitter 10,  Dead 40, Retransmit 5, TransmitDelay 1
  DelayedLSAck:     4 times
  Authentication: Not Configured
  Statistics:
      Type     tx          rx          tx-byte     rx-byte
      Unknown  0           0           0           0
      Hello    79          77          3156        3076
      DbDesc   4           5           352         500
      LSReq    2           2           200         152
      LSUpdate 9           10          1056        1188
      LSAck    5           5           460         440
      OSPF messages dropped,no authentication: 0
  Neighbor: State: Full Address: 2001:1:0:2::1 Interface: Te 2/0/2
```

9. The following example of the **show ipv6 ospf database** command with the **type-7** keyword shows detailed output for a configured NSSA.

```
device# show ipv6 ospf database type-7

Area ID         Type LSID       Adv Rtr         Seq(Hex) Age  Cksum Len   Sync
4               Typ7 1          5.5.5.5         80000001 280  58ee  60    Yes
     Bits: EF-
     Metric: 0
     Prefix Options: P,
     Referenced LSType: 0
     Prefix: 100::100/128
     Forwarding-Address: 2001:5:0:3::5
Area ID         Type LSID       Adv Rtr         Seq(Hex) Age  Cksum Len   Sync
4               Typ7 2          3.3.3.3         80000002 8    02c9  44    Yes
     Bits: EF-
     Metric: 33
     Prefix Options:
     Referenced LSType: 0
     Prefix: ::/0
     Forwarding-Address: 2001:3:0:2::3
```

10. The following example of the **show ipv6 ospf routes** command shows output for OSPFv3 routes.

```
device# show ipv6 ospf routes

Destination                    Cost       E2Cost     Tag        Flags    Dis
E2 ::/0                           1          33         0          00000027 110
Next_Hop_Router                Outgoing_Interface Adv_Router
2001:3:0:2::3                  Te 4/0/2           3.3.3.3
Destination                    Cost       E2Cost     Tag        Flags    Dis
E2 100::100/128                   1          0          0          0000000b 110
Next_Hop_Router                Outgoing_Interface Adv_Router
2001:5:0:3::5                  Te 4/0/3           5.5.5.5
```

11. The following example of the **show ipv6 ospf database as-external** command shows information about external LSAs.

```
device# show ipv6 ospf database as-external

Area ID         Type LSID       Adv Rtr         Seq(Hex) Age  Cksum Len   Sync
N/A             Extn 106        6.6.6.6         80000001 533  7993  36    Yes
     Bits: E--
     Metric: 0
     Prefix Options:
     Referenced LSType: 0
     Prefix: 4001::/64

Area ID         Type LSID       Adv Rtr         Seq(Hex) Age  Cksum Len   Sync
N/A             Extn 105        6.6.6.6         80000001 533  d230  44    Yes
     Bits: E--
     Metric: 0
     Prefix Options:
     Referenced LSType: 0
     Prefix: 2::2/128
```

12. The following example of the **show ipv6 ospf database** command shows information about different OSPFv3 LSAs.

```
device# show ipv6 ospf database
LSA Key - Rtr:Router Net:Network Inap:InterPrefix Inar:InterRouter
          Extn:ASExternal Grp:GroupMembership Typ7:Type7 Link:Link
          Iap:IntraPrefix Grc:Grace

Area ID         Type LSID       Adv Rtr         Seq(Hex) Age  Cksum Len   Sync
0               Link 145        6.6.6.6         80000002 1036 0432  56    Yes
0               Link 7          1.1.1.1         80000002 595  b6b2  56    Yes
0               Rtr  0          6.6.6.6         80000004 560  d1f4  40    Yes
0               Rtr  0          1.1.1.1         80000004 1756 ce98  40    Yes
0               Net  145        6.6.6.6         80000002 560  c899  32    Yes
0               Iap  4350       6.6.6.6         80000002 560  903a  44    Yes
1               Rtr  0          1.1.1.1         80000003 1756 0c20  24    Yes
N/A             Extn 106        6.6.6.6         80000002 560  7794  36    Yes
N/A             Extn 108        6.6.6.6         80000001 1678 9888  36    Yes
```

13. The following example of the **show ipv6 ospf database** command with the **tree** shows information about the SPF trees.

```
device# show ipv6 ospf spf tree
 SPF tree for Area 0
 +- 1.1.1.1 cost 0
     +- 6.6.6.6:145 cost 1
         +- 6.6.6.6:0 cost 1

 SPF tree for Area 1
 +- 1.1.1.1 cost 0
```

14. The following example of the **show ipv6 ospf database** command with the **table** shows information about the SPF table.

```
device# show ipv6 ospf spf table
SPF table for Area 4
  Destination           Bits Options  Cost  Nexthop                   Interface
R 3.3.3.3              ---EB V6---NR--    1  fe80::227:f8ff:feca:bbb8  Te 4/0/2
R 5.5.5.5              ---E- V6---NR--    1  fe80::205:33ff:fee5:4eda  Te 4/0/3
N 4.4.4.4[2]           ----- V6---NR--    1  ::                        Te 4/0/2
N 5.5.5.5[3]           ----- V6---NR--    1  ::                        Te 4/0/3
```

15. The following example of the **show ipv6 ospf redistribute route** command shows information about routes that the device has redistributed into OSPFv3.

```
device# show ipv6 ospf redistribute route

Id     Prefix                                    Protocol  Metric Type  Metric
105    2::2/128                                  Connect   Type-2       0
108    2001:6:4::/64                             Connect   Type-2       0
106    4001::/64                                 Connect   Type-2       0
```

16. The following example of the **show ipv6 ospf routes** command shows information about a specified OSPFv3 route.

```
device# show ipv6 ospf routes 2001:3:0:1::/64
Destination                    Cost       E2Cost     Tag        Flags    Dis
OA 2001:3:0:1::/64               2          0          0          00000003 110
Next_Hop_Router                Outgoing_Interface Adv_Router
fe80::227:f8ff:feca:bbb8       Te 4/0/2           3.3.3.3
```

Clearing OSPFv3 redistributed routes

OSPFv3 redistributed routes for a device can be cleared using a CLI command.

The **show ipv6 ospf redistribute route** command is entered to verify that there are IPv6 routes on the device that have been redistributed into OSPFv3. The **clear ipv6 ospf redistribution** command is entered to clear all OSPFv3 redistributed routes. The **show ipv6 ospf redistribute route** command is re-entered to verify that the OSPFv3 redistributed routes have been cleared.

1. Enter the **exit** command. Repeat as necessary.

```
device(config)# exit
```

   Enters privileged EXEC mode.

2. Enter the **show ipv6 ospf redistribute route** command.

```
device# show ipv6 ospf redistribute route
Id     Prefix                                    Protocol  Metric Type  Metric
34     1900:53:131::130/124                      Connect   Type-2       0
36     2001:2031::/64                            Connect   Type-2       0
37     2001:3031::/64                            Connect   Type-2       0
38     2001:3032::/64                            Connect   Type-2       0
39     2001:3033::/64                            Connect   Type-2       0
32     2001:a131:131:132::/64                    Static    Type-2       1
35     2131:131::/64                             Connect   Type-2       0
33     3001::132/128                             Connect   Type-2       0
device#
```

   Displays all IPv6 routes that the device has redistributed into OSPFv3.

3. Enter the **clear ipv6 ospf redistribution** command.

```
device# clear ipv6 ospf redistribution
```

   Clears OSPFv3 redistributed routes.

4. Enter the **show ipv6 ospf redistribute route** command.

```
device# show ipv6 ospf redistribute route
Id     Prefix                                  Protocol  Metric Type  Metric
41     1900:53:131::130/124                    Connect   Type-2       0
43     2001:2031::/64                          Connect   Type-2       0
44     2001:3031::/64                          Connect   Type-2       0
45     2001:3032::/64                          Connect   Type-2       0
46     2001:3033::/64                          Connect   Type-2       0
47     2001:a131:131:132::/64                  Static    Type-2       1
42     2131:131::/64                           Connect   Type-2       0
40     3001::132/128                           Connect   Type-2       0
device#
```

Displays all IPv6 routes that the device has redistributed into OSPFv3.

The following example clears OSPFv3 redistributed routes for a device and verifies that the OSPFv3 redistributed routes have been cleared:

```
device(config-ipv6-router-ospf-vrf)# exit
device(config-rbridge-id-122)# exit
device(config)# exit
device# show ipv6 ospf redistribute route
device# clear ipv6 ospf redistribution
device# show ipv6 ospf redistribute route
```

BGP4

BGP4 overview

Border Gateway Protocol version 4 (BGP4) is an exterior gateway protocol that performs inter-autonomous system (AS) or inter-domain routing. It peers to other BGP-speaking systems over TCP to exchange network reachability and routing information. BGP primarily performs two types of routing: inter-AS routing, and intra-AS routing. BGP peers belonging to different autonomous systems use the inter-AS routing, referred as Exterior BGP (EBGP). On the other hand, within an AS BGP can be used to maintain a consistent view of network topology, to provide optimal routing, or to scale the network.

BGP is a path vector protocol and implements this scheme on large scales by treating each AS as a single point on the path to any given destination. For each route (destination), BGP maintains the AS path and uses this to detect and prevent loops between autonomous systems.

The Open Shortest Path First (OSPF) protocol (supported in Network OS 3.0 and later) provides dynamic routing within the VCS and internal domain. However, even though OSPF suffices for most of the routing needs within the VCS, an exterior gateway protocol such as BGP is needed for inter-domain routing outside the VCS domain.

Similar to other Layer 3 protocols in Network OS, BGP is supported only in the VCS mode of operation. Each RBridge in a VCS fabric running BGP acts as an individual BGP device. BGP can form iBGP peering with other RBridges in the same VCS fabric running BGP.

BGP support

Support for BGP on Network OS platforms is for BGP4 (compliant with RFC 1771 and 4271), and provides the following:

- Connectivity from the VCS to a core/external network or cloud
- A foundation to support virtual routing and forwarding (VRF) for multi-tenancy and remote-VCS access and route distribution across VRFs
- A foundation to support VRF scaling (OSPF does not scale well with lots of VRFs)
- A foundation to support OSPF Interior Gateway Protocol (IGP) scaling needs in future

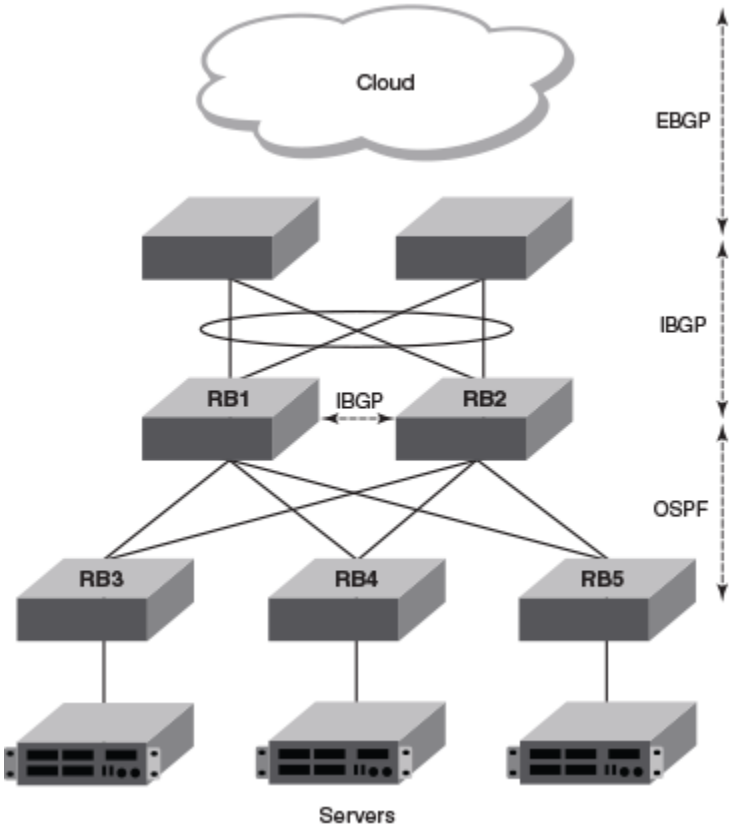
NOTE
An L3 License is required to enable BGP routing.

Deployment scenarios

BGP is typically used in a VCS Fabric at the aggregation layer and in connecting to the core. Routing bridges at the aggregation layer can either connect directly to the core, or connect through an additional device that aggregates traffic from the RBridges. The topologies below illustrate connectivity with and without an aggregator to the core. The details of these topologies are discussed in subsequent sections.

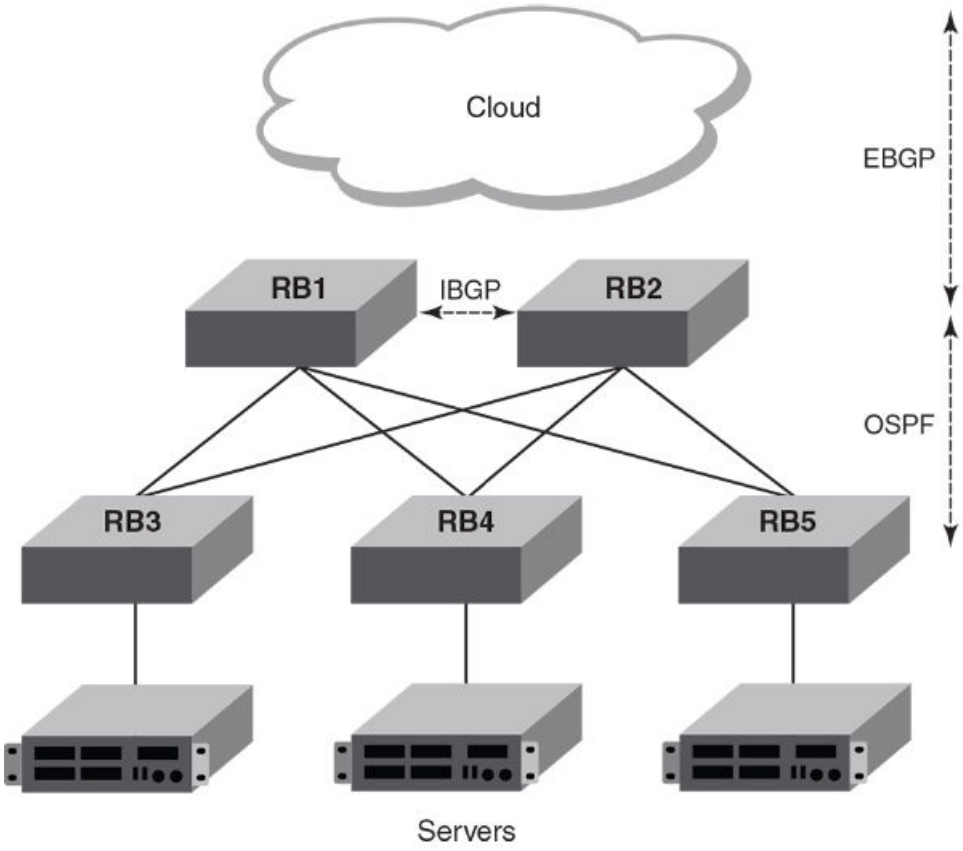
The figure below illustrates connectivity to the core through an aggregator. The RBridges use OSPF and IBGP to communicate with each other, connecting to the core aggregator through IBGP. The core aggregator connects in turn to the core through EBGP.

FIGURE 20 Connectivity to the core through an aggregator



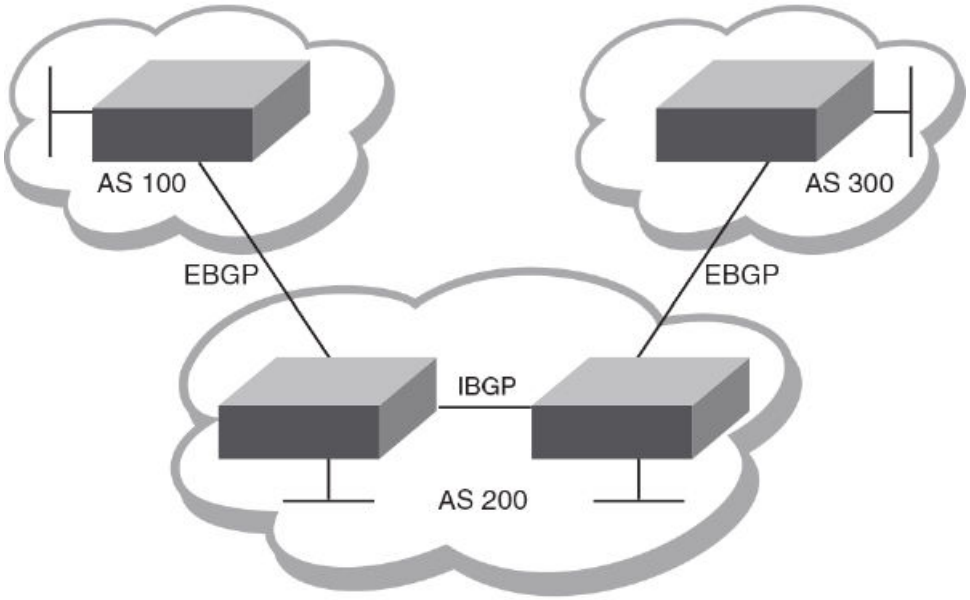
The figure below illustrates the previous topology but without an aggregator.

FIGURE 21 Connectivity to the core without an aggregator



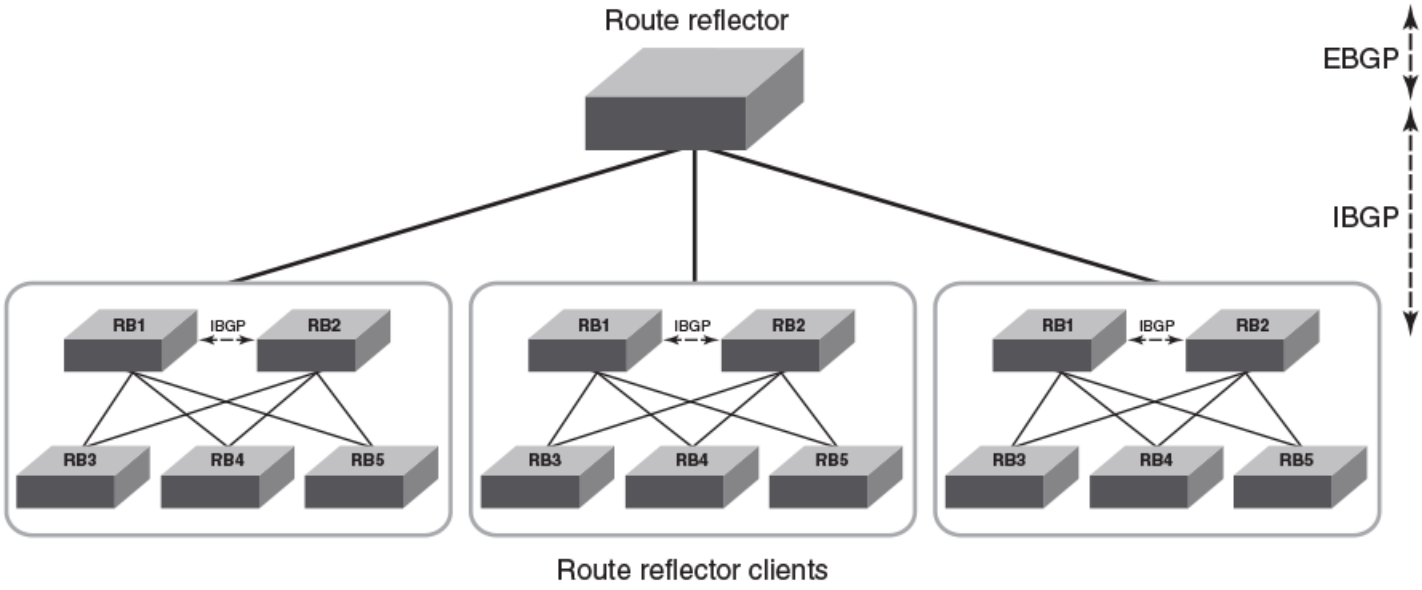
The figure below illustrates the role of BGP in communicating through multiple VCS clusters and autonomous systems.

FIGURE 22 BGP with multiple VCS clusters and autonomous systems



The figure below illustrates a BGP topology that incorporates a route-reflector server and route-reflector clients.

FIGURE 23 BGP route-reflector server and clients



BGP4 peering

Unlike OSPF or other IGP protocols, BGP4 does not have neighbor detection capability. BGP4 neighbors (or peers) must be configured manually. A device configured to run BGP4 is called a BGP "speaker." A BGP speaker connects to another speaker (either in the same or a different AS) by using a TCP connection to port 179 (the well-known BGP port), to exchange the routing information. The TCP connection is maintained throughout the peering session. While the connection between BGP peers is alive, two peers communicate by means of the following types of messages:

- OPEN
- UPDATE
- KEEPALIVE
- NOTIFICATION
- ROUTE REFRESH

BGP4 peering can be internal or external, depending on whether the two BGP peers belong to the same AS or different ASs. A BGP4 session between peers within a single AS is referred to as an Interior BGP (iBGP) session; a session between peers belonging to different ASs is referred to as an Exterior BGP (eBGP) session.

In order to establish a TCP connection between two iBGP peers, the IP reachability should be established either by means of the underlying IGP protocol (e.g. OSPF) or by means of static routes. When routes are advertised within iBGP peers, the following primary actions are taken in contrast to eBGP peering:

- Routes learned from an iBGP peer are not usually advertised to other iBGP peers, in order to prevent loops within an AS.
- Path attributes are not usually changed, in order to maintain the best path selection at other nodes within an AS.
- The AS path and next hop are not normally changed.

BGP4 message types

All BGP4 messages use a common packet header, with the following byte lengths:

Marker	Length	Type	Data
16	2	1	variable

NOTE
All values in the following tables are in bytes.

Type can be OPEN, UPDATE, NOTIFICATION, KEEPALIVE, or ROUTE-REFRESH, as described below.

OPEN message

After establishing TCP connection, BGP peers exchange OPEN message to identify each other.

Version	Autonomous System	Hold-Time	BGP Identifier	Optional Parameter Len	Optional Parameters
1	2 or 4	2	4	1	4

Version

Only BGP4 version 4 is supported.

Autonomous System

Both 2-byte and 4-byte AS numbers are supported.

KEEPALIVE and **HOLDTIME** messages

A BGP **timer** command specifies both **keep-alive** and **hold-time** operands that manage the intervals for BGP KEEPALIVE and HOLDTIME messages. The keep alive time specifies how frequently the device sends KEEPALIVE messages to its BGP4 neighbors. The hold time specifies how long the device waits for a KEEPALIVE or UPDATE message from a neighbor before concluding that the neighbor is dead. When two neighbors have different hold-time values, the lowest value is used. A hold-time value of 0 means "always consider neighbor to be active."

Refer to the *Extreme Network OS Command Reference* for more information.

BGP Identifier

Indicates the router (or device) ID of the sender. When router-id is not configured, device-id is taken from the loopback interface. Otherwise, the lowest IP address in the system is used.

Parameter List

Optional list of additional parameters used in peer negotiation.

UPDATE message

The UPDATE message is used to advertise new routes, withdraw previously advertised routes, or both.

WithdrawnRoutesLength	WithdrawnRoutes	Total PathAttributes Len	Path Attributes	NLRI
2	variable	2	variable	variable

Withdrawn Routes Length

Indicates the length of next (withdrawn routes) field. It can be 0.

Withdrawn Routes

Contains list of routes (or IP-prefix/Length) to indicate routes being withdrawn.

Total Path Attribute Len

Indicates length of next (path attributes) field. It can be 0.

Path Attributes

Indicates characteristics of the advertised path. Possible attributes: Origin, AS Path, Next Hop, MED (Multi-Exit Discriminator), Local Preference, Atomic Aggregate, Aggregator, Community, extended-Communities.

NLRI

Network Layer Reachability Information — the set of destinations whose addresses are represented by one prefix. This field contains a list of IP address prefixes for the advertised routes.

NOTIFICATION message

In case of an error that causes the TCP connection to close, the closing peer sends a notification message to indicate the type of error.

Error Code	ErrorSubcode	Error Data
1	1	variable

Error Code

Indicates the type of error, which can be one of following:

- Message header error
- Open message error
- Update message error
- Hold timer expired
- Finite state-machine error
- Cease (voluntarily)

Error Subcode

Provides specific information about the error reported.

Error Data

Contains data based on error code and subcode.

KEEPALIVE message

Because BGP does not regularly exchanges route updates to maintain a session, KEEPALIVE messages are sent to keep the session alive. A KEEPALIVE message contains just the BGP header without data field. Default KEEPALIVE time is 60 seconds and is configurable.

REFRESH message

A REFRESH message is sent to a neighbor requesting that the neighbor resend the route updates. This is useful when the inbound policy has been changed.

BGP4 attributes

BGP4 attributes are passed in UPDATE messages to describe the characteristics of a BGP path by the advertising device. At a high level, there are only two types of attributes: well-known and optional. All of the well-known attributes, as described in RFC 4271, are supported.

BGP4 best path selection algorithm

The BGP decision process is applied to the routes contained in the Routing Information Base, Incoming (RIB-In) which contains routes learned from inbound update messages. The output of the decision process is the set of routes that will be advertised to BGP speakers in local or remote autonomous systems and are stored in the Adjacency RIB, Outgoing (RIB-Out).

When multiple paths for the same route prefix are known to a BGP4 device, the device uses the following algorithm to weigh the paths and determine the optimal path for the route. The optimal path depends on various parameters, which can be modified.

Refer to the *Extreme Network OS Command Reference* for more information.

1. Verify that the next hop can be resolved by means of Interior Gateway Protocol (IGP).
2. Use the path with the largest weight.
3. If the weights are the same, prefer the path with the largest local preference.

4. Prefer the route that was self-originated locally.
5. If the local preferences are the same, prefer the path with the shortest AS-path. An AS-SET counts as 1. A confederation path length, if present, is not counted as part of the path length.

   The **as-path ignore** command disables the comparison of the AS path lengths of otherwise equal paths.

   NOTE
   This step can be skipped if the **as-path-ignore** command is configured.

6. If the AS-path lengths are the same, prefer the path with the lowest origin type. From low to high, route origin types are valued as follows:
   - IGP is lowest.
   - EGP is higher than IGP but lower than INCOMPLETE.
   - INCOMPLETE is highest.
7. If the paths have the same origin type, prefer the path with the lowest MED.

   The device compares the MEDs of two otherwise equivalent paths if and only if the routes were learned from the same neighboring AS. This behavior is called deterministic MED. Deterministic MED is always enabled and cannot be disabled.

   To ensure that the MEDs are always compared, regardless of the AS information in the paths, the **always-compare-med** command can be used. This option is disabled by default.

   The **med-missing-as-worst** command can be used to make the device regard a BGP4 route with a missing MED attribute as the least-favorable path when the MEDs of the route paths are compared.

   MED comparison is not performed for internal routes that originate within the local AS or confederation, unless the **compare-med-empty-aspath** command is configured.
8. Prefer paths in the following order:
   - Routes received through eBGP from a BGP4 neighbor outside of the confederation
   - Routes received through eBGP from a BGP4 device within the confederation *or* routes received through IBGP.
9. If all the comparisons above are equal, prefer the route with the lowest IGP metric to the BGP4 next hop. This is the closest internal path inside the AS to reach the destination.
10. If the internal paths also are the same and BGP4 load sharing is enabled, load-share among the paths. Otherwise go to Step 11.

    NOTE
    For eBGP routes, load sharing applies only when the paths are from neighbors within the same remote AS. eBGP paths from neighbors in different ASs are not compared, unless multipath multi-as is enabled.

11. If **compare-routerid** is enabled, prefer the path that comes from the BGP4 device with the lowest device ID. If a path contains originator ID attributes, then the originator ID is substituted for the router ID in the decision.
12. Prefer the path with the minimum cluster-list length.
13. Prefer the route that comes from the lowest BGP4 neighbor address.

BGP4 limitations and considerations

The following limitations and considerations apply to BGP4 support on Network OS platforms:

- There is no backward compatibility. In case of a downgrade, BGP configurations are lost.

- There is no support for nonstop routing.
- RASLogs are generated when a BGP session begins.
- RASTRACE logs are available with module ID "261 BGP".
- BGP logs are generated with module ID “BGP-1002”.

Device ID

BGP automatically calculates the device identifier it uses to specify the origin in routes it advertises. If a router-id configuration is already present in the system, then device-id is used as the router-id. Otherwise, the device first checks for a loopback interface, and the IP address configured on that interface is chosen as the device-id. However, if a loopback interface is not configured, the device-id is chosen from lowest-numbered IP interface address configured on the device. Once device-id is chosen, the device identifier is not calculated unless the IP address configured above is deleted.

BGP global mode

To enable BGP4 use the **router bgp** command in RBridge ID configuration mode.

```
device(config-rbridge-id-12)# router bgp
```

After using the **router bgp** command you enter into BGP global configuration mode.

Configurations that are not specific to address-family configuration are available in the BGP global configuration mode:

```
device(config-bgp-router)# ?
Possible completions:
  address-family                Enter Address Family command mode
  always-compare-med            Allow comparing MED from different neighbors
  as-path-ignore                Ignore AS_PATH length for best route selection
  auto-shutdown-new-neighbors   Auto shutdown new neighbor
  bfd                           Set BFD global parameters for BGP
  capability                    Set capability
  cluster-id                    Configure Route-Reflector Cluster-ID
  compare-med-empty-aspath      Allow comparing MED from different neighbors
                                even with empty as-path attribute
  compare-routerid              Compare router-id for identical BGP paths
  confederation                 Configure AS confederation parameters
  default-local-preference      Configure default local preference value
  distance                      Define an administrative distance
  enforce-first-as              Enforce the first AS for EBGP routes
  fast-external-fallover        Reset session if link to EBGP peer goes down
  install-igp-cost              Install igp cost to nexthop instead of MED
                                value as BGP route cost
  local-as                      Configure local AS number
  log-dampening-debug           Log dampening debug messages
  maxas-limit                   Impose limit on number of ASes in AS-PATH attribute
  med-missing-as-worst          Consider routes missing MED attribute as least desirable
  neighbor                      Specify a neighbor router
  timers                        Adjust routing timers
```

Configuring a local AS number

The local AS number (ASN) identifies the AS in which the BGP device resides. The following task configures the local ASN in which the device resides.

NOTE
Use well-known private ASNs in the range from 64512 through 65535 if the AS number of the organization is not known.

1. Enter the **configure terminal** command to access global configuration mode.

```
device# configure terminal
```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

```
device(config)# rbridge-id 122
```

3. Enter the **router bgp** command to enable BGP routing.

```
device(config-rbridge-id-122)# router bgp
```

4. Enter the **local-as** command to configure the autonomous system number (ASN) in which your device resides.

```
device(config-bgp-router)# local-as 1000
```

The following example configures the local ASN for a device.

```
device# configure terminal
device(config)# rbridge-id 122
device(config-rbridge-id-122)# router bgp
device(config-bgp-router)# local-as 1000
```

IPv4 unicast address family

The IPv4 unicast address family configuration level provides access to commands that allow you to configure BGP4 unicast routes. The commands that you enter at this level apply only to the IPv4 unicast address family.

BGP4 supports the IPv4 address family configuration level. This configuration is applied in the IPv4 address-family unicast submode of BGP.

```
device(config-bgp-router)# address-family ipv4 unicast
device(config-bgp-ipv4u)#
```

The commands that you can access while at the IPv4 unicast address family configuration level are also available at the IPv6 unicast address family configuration levels. Each address family configuration level allows you to access commands that apply to that particular address family only.

Where relevant, this chapter discusses and provides IPv4-unicast-specific examples.

The following configuration options are allowed under BGP IPv4 address family unicast configuration mode:

```
device(config-bgp-router)# address-family ipv4 unicast

device(config-bgp-ipv4u)# ?
Possible completions:
  aggregate-address              Configure BGP aggregate entries
  always-propagate               Allow readvertisement of best BGP routes not
                                 in IP Forwarding table
  bgp-redistribute-internal      Allow redistribution of iBGP routes into IGPs
  client-to-client-reflection    Configure client to client route reflection
  dampening                      Enable route-flap dampening
```

```
  default-information-originate   Originate Default Information
  default-metric                  Set metric of redistributed routes
  graceful-restart                Enables the BGP graceful restart capability
  maximum-paths                   Forward packets over multiple paths
  multipath                       Enable multipath for ibgp or ebgp neighbors
                                  only
  neighbor                        Specify a neighbor router
  network                         Specify a network to announce via BGP
  next-hop-enable-default         Enable default route for BGP next-hop lookup
  next-hop-recursion              Perform next-hop recursive lookup for BGP
                                  route
  redistribute                    Redistribute information from another
                                  routing protocol
  rib-route-limit                 Limit BGP rib count in routing table
  static-network                  Special network that do not depends on IGP
                                  and always treat as best route in BGP
  table-map                       Map external entry attributes into routing
                                  table
  update-time                     Configure igp route update interval
```

Neighbor configuration

For each neighbor a device is going to peer with, there must be a neighbor configuration that specifies an IP address (which must be the primary IP address of interface connection to get established) and an AS number of the neighbor. For each neighbor, you can specify a set of attributes. However, in cases where a set of neighbors share the same set of attributes, it is advisable to create a peer-group.

The neighbor configuration appears in both the global and address-family modes of BGP. The neighbor parameters that are common to all of the address families appear in BGP global configuration mode, while the parameters that are specific to an address family appear within the BGP address-family submode.

The following **running-config** excerpt illustrates the neighbor configurations that are allowed in BGP global mode and IPv4 address-family submode:

```
router bgp
  local-as 6500
  neighbor 10.231.64.10 advertisement-interval 60
  neighbor as-override
  neighbor 10.231.64.10 capability as4-enable
  neighbor 10.231.64.10 description "Example Neighbor Configuration"
  neighbor 10.231.64.10 ebgp-multihop 2
  neighbor 10.231.64.10 enforce-first-as
  neighbor 10.231.64.10 local-as 64900
  neighbor 10.231.64.10 maxas-limit in disable
  neighbor 10.231.64.10 next-hop-self always
  neighbor 10.231.64.10 password default
  neighbor 10.231.64.10 remote-as 1200
  neighbor 10.231.64.10 remove-private-as
  neighbor 10.231.64.10 shutdown generate-rib-out
  neighbor 10.231.64.10 soft-reconfiguration inbound
  neighbor 10.231.64.10 timers keep-alive 120 hold-time 240
  neighbor 10.231.64.10 update-source loopback lo0
  address-family ipv4 unicast
   neighbor 10.231.64.10 default-originate route-map test-map
   neighbor 10.231.64.10 filter-list [ 1 ] in
   neighbor 10.231.64.10 maximum-prefix 15000 teardown
   neighbor 10.231.64.10 prefix-list test-prefix in
   neighbor 10.231.64.10 route-map in test-map
   neighbor 10.231.64.10 send-community both
   neighbor 10.231.64.10 unsuppress-map test-map-2
   neighbor 10.231.64.10 weight 10
```

Configuring BGP4 neighbors

BGP4 neighbors can be configured using this procedure.

1. Enter the **configure terminal** command to access global configuration mode.

```
device# configure terminal
```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

```
device(config)# rbridge-id 122
```

3. Enter the **router bgp** command to enable BGP routing.

```
device(config-rbridge-id-122)# router bgp
```

4. Enter the **local-as** command to configure the autonomous system number (ASN) in which your device resides.

```
device(config-bgp-router)# local-as 1000
```

5. Enter the **neighbor remote-as** command, and specify an IP address, to specify the ASN in which the remote neighbor resides.

```
device(config-bgp-router)# neighbor 10.1.1.1 remote-as 1001
```

The following example configures a BGP4 neighbor.

```
device# configure terminal
device(config)# rbridge-id 122
device(config-rbridge-id-122)# router bgp
device(config-bgp-router)# local-as 1000
device(config-bgp-router)# neighbor 10.1.1.1 remote-as 1001
```

Peer groups

Neighbors having the same attributes and parameters can be grouped together by means of the **neighbor peer-group** command. You must first create a peer-group, after which you can associate neighbor IP addresses with the peer-group. All of the attributes that are allowed on a neighbor are also allowed on a peer-group.

The benefits of peer groups are:

- Simplified neighbor configuration - You can configure a set of neighbor parameters and then apply them to multiple neighbors. You do not need to configure the common parameters individually on each neighbor.
- Flash memory conservation - Using peer groups instead of individually configuring all the parameters for each neighbor requires fewer configuration commands in the startup configuration file.

You can perform the following tasks on a peer-group basis:

- Reset neighbor sessions
- Perform soft-outbound resets (the device updates outgoing route information to neighbors but does not entirely reset the sessions with those neighbors)
- Clear BGP4 message statistics
- Clear error buffers

An attribute value configured explicitly for a neighbor takes precedence over the attribute value configured for a peer-group. If neither the peer-group nor the individual neighbor has the attribute configured, the default value for the attribute is used.

For the parameters of a peer group to take effect, the peer group must be activated in the IPv4 or IPv6 address-family. By default, only IPv4 unicast address family is activated for a peer-group. A user needs to explicitly activate a peer-group in the IPv6 unicast address-family configuration mode when used with IPv6 peers.

Configuring BGP4 peer groups

A peer group can be created and neighbor IPv4 addresses can be associated with the peer group.

1. Enter the **configure terminal** command to access global configuration mode.

```
device# configure terminal
```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

```
device(config)# rbridge-id 122
```

3. Enter the **router bgp** command to enable BGP routing.

```
device(config-rbridge-id-122)# router bgp
```

4. Enter the **local-as** command to configure the autonomous system number (ASN) in which your device resides.

```
device(config-bgp-router)# local-as 1000
```

5. Enter the **neighbor** *peer-group-name* **peer-group** command to create a peer group.

```
device(config-bgp-router)# neighbor mypeergroup1 peer-group
```

6. Enter the **neighbor** *peer-group-name* **remote-as** command to specify the ASN of the peer group.

```
device(config-bgp-router)# neighbor mypeergroup1 remote-as 11
```

7. Enter the **neighbor** *ip-address* **peer-group** command to associate a neighbor with the peer group.

```
device(config-bgp-router)# neighbor 10.2.2.2 peer-group mypeergroup1
```

8. Enter the **neighbor** *ip-address* **peer-group** command to associate another neighbor with the peer group.

```
device(config-bgp-router)# neighbor 10.3.3.3 peer-group mypeergroup1
```

The following example creates a peer group and specifies two neighbors to belong to the peer group.

```
device# configure terminal
device(config)# rbridge-id 122
device(config-rbridge-id-122)# router bgp
device(config-bgp-router)# local-as 1000
device(config-bgp-router)# neighbor mypeergroup1 peer-group
device(config-bgp-router)# neighbor mypeergroup1 remote-as 11
device(config-bgp-router)# neighbor 10.2.2.2 peer-group mypeergroup1
device(config-bgp-router)# neighbor 10.3.3.3 peer-group mypeergroup1
```

Advertising the default BGP4 route

By default, a BGP device does not originate and advertise a default route using BGP4. A BGP4 default route is the IP address 0.0.0.0 and the route prefix 0 or network mask 0.0.0.0. For example, 0.0.0.0/0 is a default route. A BGP device can be configured to advertise the default IPv4 route to all BGP4 neighbors and to install that route in the local BGP4 route table.

The default route must be present in the local IPv4 route table.

1. Enter the **configure terminal** command to access global configuration mode.

```
device# configure terminal
```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

```
device(config)# rbridge-id 122
```

3. Enter the **router bgp** command to enable BGP routing.

```
device(config-rbridge-id-122)# router bgp
```

4. Enter the **address-family ipv4 unicast** command to enter IPv4 address family unicast configuration mode.

```
device(config-bgp-router)# address-family ipv4 unicast
```

5. Enter the **default-information-originate** command to advertise the default IPv4 route to all BGP4 neighbors and to install that route in the local BGP4 route table.

```
device(config-bgp-ipv4u)# default-information-originate
```

The following example enables a BGP4 device to advertise the default IPv4 route to all BGP4 neighbors and to install that route in the local BGP4 route table.

```
device# configure terminal
device(config)# rbridge-id 122
device(config-rbridge-id-122)# router bgp
device(config-bgp-router)# address-family ipv4 unicast
device(config-bgp-ipv4u)# default-information-originate
```

Four-byte AS numbers

Four-byte autonomous system numbers (ASNs) can be optionally configured on a device, peer-group, or neighbor. If this is enabled, the device announces and negotiates "AS4" capability with its neighbors.

You can configure AS4 capability to be enabled or disabled either at the BGP global level or at the neighbor or peer-group level.

You can configure AS4 capability to be enabled for a neighbor while still keeping AS4 numbers disabled at the global level, or vice-versa. The neighbor AS4 capability configuration takes precedence. If AS4 capability is not configured on the neighbor, then the peer-group configuration takes effect. The global configuration is used if AS4 capability is configured neither at the neighbor nor at the peer-group level. If a device having a 4-byte ASN tries to connect to a device that does not have AS4 support, peering will not be established.

Cooperative BGP4 route filtering

By default, the device performs all filtering of incoming routes locally, on the device itself. You can use cooperative BGP4 route filtering to cause the filtering to be performed by a neighbor before it sends the routes to the device. Cooperative filtering conserves resources by eliminating unnecessary route updates and filter processing. For example, the device can send a deny filter to a neighbor, which the neighbor uses to filter out updates before sending them to the device. The neighbor saves the resources it would otherwise use to generate the route updates, and the device saves the resources it would use to filter out the routes.

When you enable cooperative filtering, the device advertises this capability in its Open message to the neighbor when initiating the neighbor session. The Open message also indicates whether the device is configured to send filters, receive filters, or both, and the types of filters it can send or receive. The device sends the filters as Outbound Route Filters (ORFs) in route refresh messages.

To configure cooperative filtering, perform the following tasks on the device and on the BGP4 neighbor:

- Configure the filter.

  NOTE
  Cooperative filtering is currently supported only for filters configured using IP prefix lists.

- Apply the filter as an inbound filter to the neighbor.
- Enable the cooperative route filtering feature on the device. You can enable the device to send ORFs to the neighbor, to receive ORFs from the neighbor, or both. The neighbor uses the ORFs you send as outbound filters when it sends routes to the device. Likewise, the device uses the ORFs it receives from the neighbor as outbound filters when sending routes to the neighbor.
- Reset the BGP4 neighbor session to send and receive ORFs.
- Perform these steps on the other device.

  NOTE
  If the device has inbound filters, the filters are still processed even if equivalent filters have been sent as ORFs to the neighbor.

BGP4 parameters

Some parameter changes take effect immediately while others do not take full effect until the device sessions with its neighbors are reset. Some parameters do not take effect until the device is rebooted.

The following parameter changes take effect immediately:

- Enable or disable BGP4.
- Set or change the local AS.
- Add neighbors.
- Change the update timer for route changes.
- Disable or enable fast external failover.
- Specify individual networks that can be advertised.
- Change the default local preference, default information originate setting, or administrative distance.
- Enable or disable use of a default route to resolve a BGP4 next-hop route.
- Enable or disable MED (metric) comparison.
- Require the first AS in an update from an EBGP neighbor to be the neighbor AS.
- Change MED comparison parameters.
- Disable comparison of the AS-Path length.
- Enable comparison of the device ID.
- Enable next-hop recursion.
- Change the default metric.
- Disable or re-enable route reflection.
- Configure confederation parameters.
- Disable or re-enable load sharing.
- Change the maximum number of load sharing paths.

- Change other load-sharing parameters.
- Define route flap dampening parameters.
- Add, change, or negate redistribution parameters (except changing the default MED).
- Add, change, or negate route maps (when used by the **network** command or a redistribution command).
- Apply maximum AS path limit settings for UPDATE messages.
- Aggregate routes
- Add, change, or negate filter tables that affect inbound and outbound route policies.

The following parameter changes take effect only after the BGP4 sessions on the device are cleared, or reset using the "soft" clear option:

- Change the Hold Time or Keep Alive Time.
- Apply maximum AS path limit settings to the RIB.

The following parameter change takes effect only after you disable and then re-enable redistribution:

- Change the default MED (metric).

Route redistribution

The redistribution of static, connected, and OSPF routes into BGP is supported. Similarly, routes learned through BGP can also be redistributed into OSPF.

An optional route-map can be specified, and this map will be consulted before routes are added to BGP. Management routes are not redistributed.

Redistributing routes into BGP4

Various routes can be redistributed into BGP. This tasks redistributes connected routes into BGP4.

1. Enter the **configure terminal** command to access global configuration mode.

```
device# configure terminal
```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

```
device(config)# rbridge-id 122
```

3. Enter the **router bgp** command to enable BGP routing.

```
device(config-rbridge-id-122)# router bgp
```

4. Enter the **address-family ipv4 unicast** command to enter BGP IPv4 address family unicast configuration mode.

```
device(config-bgp-router)# address-family ipv4 unicast
```

5. Enter the **redistribute** command using the **connected** keyword to redistribute connected routes.

```
device(config-bgp-ipv4u)# redistribute connected
```

The following example redistributes connected routes into BGP4.

```
device# configure terminal
device(config)# rbridge-id 122
device(config-rbridge-id-122)# router bgp
device(config-bgp-router)# address-family ipv4 unicast
device(config-bgp-ipv4u)# redistribute connected
```

Advertised networks

As previously described, you can advertise routes into BGP by redistributing static, connected, or OSPF routes.

However, you can explicitly specify routes to be advertised by BGP4 by using the **network** command in IPv4 address-family unicast configuration mode.

With the exception of static network routes, the routing table must have this route already installed before BGP4 can advertise this route. You can also specify a route to be local. If the same route is received by means of eBGP, the local IGP route will be preferred. You can also specify a weight that the device adds to routes that are received from the specified BGP neighbor. BGP4 prefers larger weights over smaller weights.

Refer to the *Extreme Network OS Command Reference*for configuration examples and more information.

Importing routes into BGP4

Routes can be explicitly specified for advertisement by BGP.

With the exception of static network routes, the routes imported into BGP4 must first exist in the IPv4 unicast route table.

1. Enter the **configure terminal** command to access global configuration mode.

   ```
   device# configure terminal
   ```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

   ```
   device(config)# rbridge-id 122
   ```

3. Enter the **router bgp** command to enable BGP routing.

   ```
   device(config-rbridge-id-122)# router bgp
   ```

4. Enter the **neighbor remote-as** command to specify the ASN in which the remote neighbor resides.

   ```
   device(config-bgp-router)# neighbor 10.2.2.2 remote-as 1001
   ```

5. Enter the **address-family ipv4 unicast** command to enter address family IPv4 unicast configuration mode.

   ```
   device(config-bgp-router)# address-family ipv4 unicast
   ```

6. Enter the **network** command and specify a *network/mask* to import the specified prefix into the BGP4 database.

   ```
   device(config-bgp-ipv4u)# network 10.1.1.1/32
   ```

The following example imports the 10.1.1.1/32 prefix in to the BGP4 database for advertising.

```
device# configure terminal
device(config)# rbridge-id 122
device(config-rbridge-id-122)# router bgp
device(config-bgp-router)# neighbor 10.2.2.2 remote-as 1001
device(config-bgp-router)# address-family ipv4 unicast
device(config-bgp-ipv4u)# network 10.1.1.1/32
```

Static networks

Before advertising any route, BGP checks for its existence in the routing table. BGP can be configured to advertise a stable route that does not depend on its existence in the routing table.

This allows you to configure a static network in BGP4, creating a stable BGP4 network in the core. While a route configured with this feature will never flap unless it is manually deleted, a "static" BGP4 network will not interrupt the normal BGP4 decision process on other learned routes being installed into the Routing Table Manager (RTM). Consequently, when there is a route that can be resolved, it will be installed into the RTM.

When the configured route is lost, BGP installs the "null0" route in the routing table. Later, when the route is resolved, the null0 route is removed.

Configuring a static network

BGP can be configured to advertise a stable route that does not depend on its existence in the routing table. The following task configures a static network and sets the administrative distance.

1. Enter the **configure terminal** command to access global configuration mode.

   ```
   device# configure terminal
   ```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

   ```
   device(config)# rbridge-id 122
   ```

3. Enter the **router bgp** command to enable BGP routing.

   ```
   device(config-rbridge-id-122)# router bgp
   ```

4. Enter the **local-as** command to configure the autonomous system number (ASN) in which your device resides.

   ```
   device(config-bgp-router)# local-as 1000
   ```

5. Enter the **address-family ipv4 unicast** command to enter IPv4 address family configuration mode.

   ```
   device(config-bgp-router)# address-family ipv4 unicast
   ```

6. Enter the **static-network** command with the **distance** parameter, and specify a value, to configure a static network and set an administrative distance.

   ```
   device(config-bgp-ipv4u)# static-network 10.11.12.0/32 distance 300
   ```

The following example configures a static network and sets an administrative distance of 300.

```
device# configure terminal
device(config)# rbridge-id 122
device(config-rbridge-id-122)# router bgp
device(config-bgp-router)# local-as 1000
device(config-bgp-router)# address-family ipv4 unicast
device(config-bgp-ipv4u)# static-network 10.11.12.0/32 distance 300
```

Route reflection

A BGP device can act as a route-reflector client or as a route reflector. You can configure a BGP peer as a route-reflector client from the device that is going to reflect the routes and act as the route reflector using the **neighbor route-reflector-client** command.

When there is more than one route reflector, they should all belong to the same cluster. By default, the value for **cluster-id** is used as the device ID. The device ID can be changed using the **cluster-id** command.

The route-reflector server reflects the routes as follows:

- Routes from the client are reflected to the client as well as to nonclient peers.
- Routes from nonclient peers are reflected only to client peers.

If route-reflector clients are connected in a full iBGP mesh, you can disable client-to-client reflection on the route reflector using the **no client-to-client-reflection** command.

A BGP device advertises only those routes that are preferred ones and are installed into the Routing Table Manager (RTM). When a route cannot be installed into the RTM because the routing table is full, the route reflector may not reflect that route. In cases where the route reflector is not placed directly in the forwarding path, you can configure the route reflector to reflect routes even though those routes are not in the RTM using the **always-propagate** command.

Configuring a cluster ID for a route reflector

The cluster ID can be changed if there is more than one route reflector, so that all route reflectors belong to the same cluster.

1. Enter the **configure terminal** command to access global configuration mode.

   ```
   device# configure terminal
   ```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

   ```
   device(config)# rbridge-id 122
   ```

3. Enter the **router bgp** command to enable BGP routing.

   ```
   device(config-rbridge-id-122)# router bgp
   ```

4. Enter the **local-as** command to configure the autonomous system number (ASN) in which your device resides.

   ```
   device(config-bgp-router)# local-as 1000
   ```

5. Enter the **cluster-id** command and specify a value to change the cluster ID of a device from the default device ID.

   ```
   device(config-bgp-router)# cluster-id 321
   ```

The following example changes the cluster ID of a device from the default device ID to 321.

```
device# configure terminal
device(config)# rbridge-id 122
device(config-rbridge-id-122)# router bgp
device(config-bgp-router)# cluster-id 321
```

Configuring a route reflector client

A BGP peer can be configured as a route reflector client.

1. Enter the **configure terminal** command to access global configuration mode.

   ```
   device# configure terminal
   ```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

   ```
   device(config)# rbridge-id 122
   ```

3. Enter the **router bgp** command to enable BGP routing.

   ```
   device(config-rbridge-id-122)# router bgp
   ```

4. Enter the **local-as** command to configure the autonomous system number (ASN) in which your device resides.

   ```
   device(config-bgp-router)# local-as 1000
   ```

5. Enter the **address-family ipv4 unicast** command to enter IPv4 address family configuration mode.

   ```
   device(config-bgp-router)# address-family ipv4 unicast
   ```

6. Enter the **neighbor route-reflector-client** command to configure a specified neighbor to be a route reflector client.

   ```
   device(config-bgp-ipv4u)# neighbor 10.1.1.1 route-reflector-client
   ```

The following example configures a neighbor with the IPv4 address 10.1.1.1 to be a route reflector client.

```
device# configure terminal
device(config)# rbridge-id 122
device(config-rbridge-id-122)# router bgp
device(config-bgp-router)# local-as 1000
device(config-bgp-router)# address-family ipv4 unicast
device(config-bgp-ipv4u)# neighbor 10.1.1.1 route-reflector-client
```

Route flap dampening

A route flap is a change in the state of a route, from up to down or down to up. A route state change causes changes in the route tables of the devices that support the route.

Frequent route state changes can cause Internet instability and add processing overhead to the devices that support the route. Route flap dampening helps reduce the impact of route flap by changing the way a BGP4 device responds to route state changes. When route flap dampening is configured, the device suppresses unstable routes until the number of route state changes drops enough to meet an acceptable degree of stability.

The route flap dampening mechanism is based on penalties. When a route exceeds a configured penalty value, the device stops using that route and stops advertising it to other devices. The mechanism also allows route penalties to reduce over time if route stability improves.

Aggregating routes advertised to BGP neighbors

A device can be configured to aggregate routes in a range of networks into a single IP prefix.

1. Enter the **configure terminal** command to access global configuration mode.

```
device# configure terminal
```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

```
device(config)# rbridge-id 122
```

3. Enter the **router bgp** command to enable BGP routing.

```
device(config-rbridge-id-122)# router bgp
```

4. Enter the **address-family ipv4 unicast** command to enter BGP address family IPv4 unicast configuration mode.

```
device(config-bgp-router)# address-family ipv4 unicast
```

5. Enter the **aggregate-address** command to aggregate the routes from a range of networks into a single network prefix.

```
device(config-bgp-ipv4u)# aggregate-address 10.1.1.1/32
```

The following example enables a BGP4 device to advertise the default route and send the default route to a specified neighbor.

```
device# configure terminal
device(config)# rbridge-id 122
device(config-rbridge-id-122)# router bgp
device(config-bgp-router)# address-family ipv4 unicast
device(config-bgp-ipv4u)# aggregate-address 10.1.1.1/32
```

Advertising the default BGP4 route

By default, a BGP device does not originate and advertise a default route using BGP4. A BGP4 default route is the IP address 0.0.0.0 and the route prefix 0 or network mask 0.0.0.0. For example, 0.0.0.0/0 is a default route. A BGP device can be configured to advertise the default IPv4 route to all BGP4 neighbors and to install that route in the local BGP4 route table.

The default route must be present in the local IPv4 route table.

1. Enter the **configure terminal** command to access global configuration mode.

```
device# configure terminal
```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

```
device(config)# rbridge-id 122
```

3. Enter the **router bgp** command to enable BGP routing.

```
device(config-rbridge-id-122)# router bgp
```

4. Enter the **address-family ipv4 unicast** command to enter IPv4 address family unicast configuration mode.

```
device(config-bgp-router)# address-family ipv4 unicast
```

5. Enter the **default-information-originate** command to advertise the default IPv4 route to all BGP4 neighbors and to install that route in the local BGP4 route table.

```
device(config-bgp-ipv4u)# default-information-originate
```

The following example enables a BGP4 device to advertise the default IPv4 route to all BGP4 neighbors and to install that route in the local BGP4 route table.

```
device# configure terminal
device(config)# rbridge-id 122
device(config-rbridge-id-122)# router bgp
device(config-bgp-router)# address-family ipv4 unicast
device(config-bgp-ipv4u)# default-information-originate
```

Advertising the default BGP4 route to a specific neighbor

A BGP device can be configured to advertise the default IPv4 route to a specific neighbor.

1. Enter the **configure terminal** command to access global configuration mode.

```
device# configure terminal
```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

```
device(config)# rbridge-id 122
```

3. Enter the **router bgp** command to enable BGP routing.

```
device(config-rbridge-id-122)# router bgp
```

4. Enter the **local-as** command to configure the autonomous system number (ASN) in which your device resides.

```
device(config-bgp-router)# local-as 1000
```

5. Enter the **address-family ipv4 unicast** command to enter IPv4 address family configuration mode.

```
device(config-bgp-router)# address-family ipv4 unicast
```

6. Enter the **neighbor default-originate** command and specify an IP address to enable the BGP4 device to advertise the default IPv4 route to a specific neighbor.

```
device(config-bgp-ipv4u)# neighbor 10.4.4.4 default-originate
```

The following example enables a BGP4 device to advertise the default IPv4 route to a specific neighbor.

```
device# configure terminal
device(config)# rbridge-id 122
device(config-rbridge-id-122)# router bgp
device(config-bgp-router)# local-as 1000
device(config-bgp-router)# address-family ipv4 unicast
device(config-bgp-ipv4u)# neighbor 10.4.4.4 default-originate
```

Multipath load sharing

Unlike IGP, BGP does not perform multipath load sharing by default. Therefore, the maximum number of paths across which BGP can balance the traffic is set to 1 by default. You can change this value by using the **maximum-paths** command.

By default, when BGP4 multipath load sharing is enabled, both iBGP and eBGP paths are eligible for load sharing, while paths from different neighboring autonomous systems are not eligible. You can change load sharing to apply only to iBGP or eBGP paths, or to support load sharing among paths from different neighboring autonomous systems.

If the maximum-path value must be picked up from the **ip load-sharing** configuration on the router, use the following:

```
sw0(config-bgp-ipv4u)# maximum-paths use-load-sharing
```

NOTE

The **ip load-sharing** command has been deprecated. Note the following use cases with respect to legacy configurations:

1. If the **ip load-sharing** command has been used in a previous release and maximum paths **has not** been configured, then load balancing is influenced only for static routes and load balancing for the routing protocol is not affected.
2. If the **ip load-sharing** command has been used in a previous release and maximum paths **has** been configured, then the lowest configured number of paths takes precedence.
3. With the current release, if only the **ip load-sharing** command has been used with an existing configuration and maximum paths **has not** been configured, then there is no effect on load sharing for the routing protocol.
4. With the current release, if only the **ip load-sharing** command has been used with an existing configuration and maximum paths **has** been configured, then the lowest configured number of paths takes precedence. The number of maximum paths can still be adjusted, by means of the **maximum-paths** command, prior to an upgrade to the current release.

Specifying the weight added to received routes

The weight that the device adds to received routes can be specified. The following task changes the weight from the default for routes that are received from a specified BGP neighbor.

1. Enter the **configure terminal** command to access global configuration mode.

   ```
   device# configure terminal
   ```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

   ```
   device(config)# rbridge-id 122
   ```

3. Enter the **router bgp** command to enable BGP routing.

   ```
   device(config-rbridge-id-122)# router bgp
   ```

4. Enter the **local-as** command to configure the autonomous system number (ASN) in which your device resides.

   ```
   device(config-bgp-router)# local-as 65520
   ```

5. Enter the **address-family ipv4 unicast** command to enter address family IPv4 unicast configuration mode.

   ```
   device(config-bgp-router)# address-family ipv4 unicast
   ```

6. Enter the **neighbor weight** command and specify an *ip address* and a weight value to specify a weight that the device adds to routes that are received from the specified BGP4 neighbor.

```
device(config-bgp-ipv4u)# neighbor 10.11.12.13 weight 100
```

The following example specifies a weight of 100 that the device adds to routes that are received from the specified BGP4 neighbor.

```
device# configure terminal
device(config)# rbridge-id 122
device(config-rbridge-id-122)# router bgp
device(config-bgp-router)# local-as 65520
device(config-bgp-router)# address-family ipv4 unicast
device(config-bgp-ipv4u)# neighbor 10.11.12.13 weight 100
```

Using the IPv4 default route as a valid next hop for a BGP4 route

In certain cases, such as when a device is acting as an edge device, it can be configured to use the default route as a valid next hop.

By default, a device does not use a default route to resolve a BGP4 next-hop route. If the IPv4 route lookup for the BGP4 next-hop does not result in a valid IGP route (including static or direct routes), the BGP4 next-hop is considered to be unreachable and the BGP4 route is not used. You can configure the device to use the default route as a valid next hop.

1. Enter the **configure terminal** command to access global configuration mode.

```
device# configure terminal
```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

```
device(config)# rbridge-id 122
```

3. Enter the **router bgp** command to enable BGP routing.

```
device(config-rbridge-id-122)# router bgp
```

4. Enter the **address-family ipv4 unicast** command to enter IPv4 address family unicast configuration mode.

```
device(config-bgp-router)# address-family ipv4 unicast
```

5. Enter the **next-hop-enable-default** command to configure the device to use the default route as a valid next hop.

```
device(config-bgp-ipv4u)# next-hop-enable-default
```

The following example configures a BGP4 device to use the default route as a valid next hop.

```
device# configure terminal
device(config)# rbridge-id 122
device(config-rbridge-id-122)# router bgp
device(config-bgp-router)# address-family ipv4 unicast
device(config-bgp-ipv4u)# next-hop-enable-default
```

Adjusting defaults to improve routing performance

The following examples illustrate a variety of options for enabling and fine-tuning route flap dampening.

The following example enables default dampening as an address-family function.

```
device# configure terminal
device(config)# rbridge-id 10
device(config-rbridge-id-10)# router bgp
device(config-bgp-router)# address-family ipv4 unicast
device(config-bgp-ipv4u)# dampening
```

The following example changes all dampening values.

```
device# configure terminal
device(config)# rbridge-id 10
device(config-rbridge-id-10)# router bgp
device(config-bgp-router)# address-family ipv4 unicast
device(config-bgp-ipv4u)# dampening 20 200 2500 40
```

Next-hop recursion

For each BGP4 route learned, the device performs a route lookup to obtain the IPv4 address of the next hop for the route. A BGP4 route is eligible for addition in the IPv4 route table only if the following conditions are true:

- The lookup succeeds in obtaining a valid next-hop IPv4 address for the route.
- The path to the next-hop IP address is an IGP path or a static route path.

By default, only one lookup is performed for the next-hop IPv4 address for the BGP4 route. If the next hop lookup does not result in a valid next hop IPv4 address, or the path to the next hop IPv4 address is a BGP4 path, the BGP4 route destination is considered unreachable. The route is not eligible to be added to the IPv4 route table.

The BGP4 route table can contain a route with a next hop IPv4 address that is not reachable through an IGP route, even though the device can reach a hop farther away through an IGP route. This can occur when the IGPs do not learn a complete set of IGP routes, so the device learns about an internal route through iBGP instead of through an IGP. In this case, the IPv4 route table does not contain a route that can be used to reach the BGP4 route destination.

When next-hop recursion is enabled, if the lookup for the next hop IP address results in an iBGP path that originated in the same AS, then the next hop is considered as resolved and BGP4 depended routes are eligible for addition in the IPV4 route table.

Enabling next-hop recursion

Next hop recursion can be enabled so that a device can find the IGP route to the next hop gateway for a BGP4 route.

1. Enter the **configure terminal** command to access global configuration mode.

   ```
   device# configure terminal
   ```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

   ```
   device(config)# rbridge-id 122
   ```

3. Enter the **router bgp** command to enable BGP routing.

   ```
   device(config-rbridge-id-122)# router bgp
   ```

4. Enter the **address-family ipv4 unicast** commandto enter IPv4 address family configuration mode.

```
device(config-bgp-router)# address-family ipv4 unicast
```

5. Enter the **next-hop-recursion** command to enable recursive next hop lookups.

```
device(config-bgp-ipv4u)# next-hop-recursion
```

The following example enables recursive next hop lookups.

```
device# configure terminal
device(config)# rbridge-id 122
device(config-rbridge-id-122)# router bgp
device(config-bgp-router)# address-family ipv4 unicast
device(config-bgp-ipv4u)# next-hop-recursion
```

Route filtering

The following route filters are supported:

- AS-path filter
- Community filter
- Prefix list
- Route map
- Table map

NOTE
Support for access lists in route filtering is not available, and has been replaced by prefix-list filtering. BGP does not use community and extended-community filters directly. Rather, it uses them indirectly through route-map filtering by means of the **route-map** command.

BGP VRF route filters

Filtering based on route-map policies can be added to the EVPN IPv4 and IPv6 prefix routes that are imported to or exported from VRFs.

The IPv4/IPv6 prefix routes of a given VRF instance in the BGP routing table can be imported from or exported to the EVPN routing table by means of the **route target (EVPN)** command.

Import routes

The **route-target (VRF)** command is used to import the corresponding *x:y* route-target extended-community-matched IPv4 or IPv6 prefix routes that are learned from the EVPN table to the VRF BGP table. The **import map evpn** command is then used to apply the filtering policies to all those routes before they are imported. Filtering is also supported for EVPN host routes (ARP/ND routes imported from EVPN to BGP VRF as /32 or /128 prefix routes). When the VRF import map is configured or removed, or additions, modifications, or deletions are made to the corresponding route-map, the **clear bgp evpn neighbor all soft in** command is effectively applied automatically to all EVPN neighbors.

Export routes

The **route-target (VRF)** command is also used to export IPv4 or IPv6 prefix routes from the VRF BGP table to the EVPN table with a route-target extended-community value of *x:y*. The **export map evpn** command is then used to apply the filtering policies to all the exported routes. When the VRF export map is configured or removed, or additions, modifications, or deletions are made to the corresponding route-map, the EVPN table is refreshed for the appropriate address family, IPv4 or IPv6.

Regular-expression-based extended-community match filters

The **ip extcommunity-list** command now supports regular-expression matching of route target or site of origin values to filter VRF routes. The previous implementation supported the **rt** (route target) and **soo** (site of origin) keywords, based on standard extended-community match filters where the **rt** and **soo** values are matched exactly. The range of standard extended-community match filters is from 1 through 99. The range of expanded extended-community match filters is from 100 through 500.

A regular expression is entered as part of a command and is a pattern made up of symbols, letters, and numbers that represent an input string for matching (or sometimes not matching). Matching the string to the specified pattern is called *pattern matching*.

Pattern matching either succeeds or fails. If a regular expression can match two different parts of an input string, it will match the earliest part first.

NOTE
For details, refer to "BGP regular expression pattern-matching characters" in this document.

Example standard and expanded extended-community configuration

The following illustrates an extended-community list configuration that includes both standard and expanded match filters.

```
device# configure terminal
device(config)# rbridge-id 1
device(config-rbridge-id-1)# ip extcommunity-list 1 permit rt 1:1 rt 1:2 rt 1:3 rt 1:4
device(config-rbridge-id-1)# ip extcommunity-list 2 permit rt 1:9 soo 1:7
device(config-rbridge-id-1)# ip extcommunity-list 100 permit 6513:*
device(config-rbridge-id-1)# ip extcommunity-list 101 permit 1:[2-6]
device(config-rbridge-id-1)# route-map map1 permit 1
device(config-routemap-map1/permit/1)# match extcommunity 1 2 100 101
```

The incoming route has to match either one of the four extended community lists: 1 or 2 or 100 or 101 (logical OR).

The incoming route has to match all the values within any one of the four extended community lists configured above (logical AND).

Using BGP VRF route filters

This task illustrates the configuration of BGP VRF import and export route filters in conjunction with a standard and an expanded extended-community list.

1. Enter global configuration mode.

```
device# configure terminal
device(config)#
```

2. Enter RBridge ID configuration mode.

```
device(config)# rbridge-id 1
device(config-rbridge-id-1)#
```

3. Specify a VRF instance and enter VRF configuration mode.

```
device(config-rbridge-id-1)# vrf red
device(config-vrf-red)#
```

4. Enter VRF address-family IPv4 unicast configuration mode.

```
device(config-vrf-red)# address-family ipv4 unicast
device(vrf-red-ipv4-unicast)#
```

NOTE
Use the **address-family ipv6 unicast** command for IPv6 configurations.

5. Use the **import map evpn** command to apply the route map "myimportmap" to imported prefixes.

```
device(vrf-red-ipv4-unicast)# import map myimportmap evpn
```

6. Use the **export map evpn** command to apply the route map "myexportmap" to exported prefixes.

```
device(vrf-red-ipv4-unicast)# export map myexportmap evpn
```

7. Return to RBridge ID configuration mode and configure a standard extended-community list (range is from 1 through 99), using the **rt** (route target) and **soo** (site of origin) keywords.

```
device(config-rbridge-id-1)# ip extcommunity-list 1 permit rt 1:9 soo 1:7
```

8. Optionally, configure an expanded extended-community list by specifying a regular-expression match pattern.

```
device(config-rbridge-id-1)# ip extcommunity-list 100 permit 6513:*
```

NOTE
For details, refer to "BGP regular expression pattern-matching characters" in this document.

BGP regular expression pattern-matching characters

The following table illustrates the functions of BGP regular expression pattern-matching characters and illustrates their use.

NOTE
The **ip-extcommunity-list** command now supports a range of extended instances, from 100 through 500, beyond the standard range of 1 through 99.

TABLE 13 BGP regular expression pattern-matching characters

Regular expression character	Function	Examples
.	Matches any single character.	0.0 matches 0x0 and 020 t..t matches strings such as test, text, and tart
\	Matches the character following the backslash. Also matches (escapes) special characters.	172\.1\.. matches 172.1.10.10 but not 172.12.0.0 \. allows a period to be matched as a period
[ ]	Matches the characters or a range of characters separated by a hyphen, within left and right square brackets.	[02468a-z] matches 0, 4, and w, but not 1, 9, or K
^	Matches the character or null string at the beginning of an input string.	^123 matches 1234, but not 01234

TABLE 13 BGP regular expression pattern-matching characters (continued)

Regular expression character	Function	Examples
?	Matches zero or one occurrence of the pattern. (Precede the question mark with Ctrl-V sequence to prevent it from being interpreted as a help command.)	ba?b matches bb and bab
$	Matches the character or null string at the end of an input string.	123$ matches 0123, but not 1234
*	Matches zero or more sequences of the character preceding the asterisk. Also acts as a wildcard for matching any number of characters.	5* matches any occurrence of the number 5 including none 18\..* matches the characters 18. and any characters that follow 18.
+	Matches one or more sequences of the character preceding the plus sign.	8+ requires there to be at least one number 8 in the string to be matched
() []	Nest characters for matching. Separate endpoints of a range with a dash (-).	(17)* matches any number of the two-character string 17 ([A-Za-z][0-9])+ matches one or more instances of letter-digit pairs: b8 and W4, as examples
|	Concatenates constructs. Matches one of the characters or character patterns on either side of the vertical bar.	A(B|C)D matches ABD and ACD, but not AD, ABCD, ABBD, or ACCD
_	Replaces a long regular expression list by matching a comma (,), left brace ({), right brace (}), the beginning of the input string, the end of the input string, or a space.	The characters _1300_ can match any of the following strings: ^1300$ ^1300space space1300 {1300, ,1300, {1300} ,1300,

Timers

The keep alive time specifies how frequently the device sends KEEPALIVE messages to its BGP4 neighbors. The hold time specifies how long the device waits for a KEEPALIVE or UPDATE message from a neighbor before concluding that the neighbor is dead. When the device concludes that a BGP4 neighbor is dead, the device ends the BGP4 session and closes the TCP connection to the neighbor.

A hold-time value of 0 means that the device waits indefinitely for messages from a neighbor without tearing down the session.

Enabling BGP4 in a non-default VRF

When BGP4 is enabled in a non-default VRF instance, the device enters BGP address-family IPv4 unicast VRF configuration mode. Several commands can then be accessed that allow the configuration of BGP4 for the non-default VRF instance.

A non-default VRF instance has been configured.

1. Enter the **configure terminal** command to access global configuration mode.

```
device# configure terminal
```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

```
device(config)# rbridge-id 122
```

3. Enter the **router bgp** command to enable BGP routing globally.

```
device(config-rbridge-id-122)# router bgp
```

4. Enter the **address-family ipv4 unicast** command with the **vrf** parameter, and specify a VRF name, to enter BGP address-family IPv4 unicast VRF configuration mode.

```
device(config-bgp-router)# address-family ipv4 unicast vrf red
```

The following example enables BGP address-family IPv4 unicast VRF configuration mode where several commands can be accessed that allow the configuration of BGP4 for the non-default VRF instance..

```
device# configure terminal
device(config)# rbridge-id 122
device(config-rbridge-id-122)# router bgp
device(config-bgp-router)# address-family ipv4 unicast vrf red
```

BGP4 outbound route filtering

The BGP4 Outbound Route Filtering Capability (ORF) feature is used to minimize the number of BGP updates sent between BGP peers.

When the ORF feature is enabled, unwanted routing updates are filtered out, reducing the amount of system resources required for generating and processing routing updates. The ORF feature is enabled through the advertisement of ORF capabilities to peer routers. The locally configured BGP4 inbound prefix filters are sent to the remote peer so that the remote peer applies the filter as an outbound filter for the neighbor.

The ORF feature can be configured with send and receive ORF capabilities. The local peer advertises the ORF capability in send mode, indicating that it will accept a prefix list from a neighbor and apply the prefix list to locally configured ORFs. The local peer exchanges the ORF capability in send mode with a remote peer for a prefix list that is configured as an inbound filter for that peer locally. The remote peer only sends the first update once it receives a ROUTEREFRESH request or BGP ORF with IMMEDIATE from the peer. The local and remote peers exchange updates to maintain the ORF on each router.

Configuring BGP4 outbound route filtering

The BGP4 Outbound Route Filtering (ORF) prefix list capability can be configured in receive mode, send mode, or both send and receive modes, minimizing the number of BGP updates exchanged between BGP peers.

1. Enter the **configure terminal** command to access global configuration mode.

```
device# configure terminal
```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

```
device(config)# rbridge-id 122
```

3. Enter the **router bgp** command to enable BGP routing.

```
device(config-rbridge-id-122)# router bgp
```

4. Enter the **address-family ipv4 unicast** command to enter IPv4 address family configuration mode.

```
device(config-bgp-router)# address-family ipv4 unicast
```

5. Enter the **neighbor prefix-list** command and specify the **in** keyword to filter the incoming route updates from a specified BGP neighbor.

```
device(config-bgp-ipv4u)# neighbor 10.1.2.3 prefix-list myprefixlist in
```

6. Do one of the following:
   - Enter the **neighbor capability orf prefixlist** command and specify the **send** keyword to advertise ORF send capabilities.

     ```
     device(config-bgp-ipv4u)# neighbor 10.1.2.3 capability orf prefixlist send
     ```

   - Enter the **neighbor capability orf prefixlist** command and specify the **receive** keyword to advertise ORF receive capabilities.

     ```
     device(config-bgp-ipv4u)# neighbor 10.1.2.3 capability orf prefixlist receive
     ```

   - Enter the **neighbor capability orf prefixlist** command to configure ORF capability in both send and receive modes.

     ```
     device(config-bgp-ipv4u)# neighbor 10.1.2.3 capability orf prefixlist
     ```

The following example configures ORF in receive mode.

```
device# configure terminal
device(config)# rbridge-id 122
device(config-rbridge-id-122)# router bgp
device(config-bgp-router)# address-family ipv4 unicast
device(config-bgp-ipv4u)# neighbor 10.1.2.3 capability orf prefixlist receive
```

The following example configures ORF in send mode.

```
device# configure terminal
device(config)# rbridge-id 122
device(config-rbridge-id-122)# router bgp
device(config-bgp-router)# address-family ipv4 unicast
device(config-bgp-ipv4u)# 10.1.2.3 prefix-list myprefixlist in
device(config-bgp-ipv4u)# 10.1.2.3 capability orf prefixlist send
```

The following example configures ORF in both send and receive modes.

```
device# configure terminal
device(config)# rbridge-id 122
device(config-rbridge-id-122)# router bgp
device(config-bgp-router)# address-family ipv4 unicast
device(config-bgp-ipv4u)# neighbor 10.1.2.3 prefix-list myprefixlist in
device(config-bgp-ipv4u)# neighbor 10.1.2.3 capability orf prefixlist
```

Enabling BGP4 cooperative route filtering

You can use cooperative BGP4 route filtering to cause the filtering to be performed by a neighbor before it sends the routes to the device, conserving resources by eliminating unnecessary route updates and filter processing. The following task enables cooperative route filtering.

1. Enter the **configure terminal** command to access global configuration mode.

   ```
   device# configure terminal
   ```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

   ```
   device(config)# rbridge-id 122
   ```

3. Enter the **ip prefix-list** command to to configure the IP prefix list instance.

   ```
   device(config-rbridge-id-122)# ip prefix-list Routesfrom10234 deny 10.20.0.0/24
   device(config-rbridge-id-122)# ip prefix-list Routesfrom10234 permit 10.0.0.0/0 le 32
   ```

4. Enter the **router bgp** command to enable BGP routing.

```
device(config-rbridge-id-122)# router bgp
```

5. Enter the **address-family ipv4 unicast** commandto enter IPv4 address family configuration mode.

```
device(config-bgp-router)# address-family ipv4 unicast
```

6. Enter the **neighbor prefix-list** command with the **in** parameter and specify a prefix-list to filter the incoming route updates from the specified BGP neighbor.

```
device(config-bgp-ipv4u)# neighbor 10.2.3.4 prefix-list Routesfrom1234 in
```

7. Enter the **capability orf prefixlist** command with the **send** parameter to enable the ORF prefix list capability in send mode.

```
device(config-bgp-ipv4u)# neighbor 10.2.3.4 capability orf prefixlist send
```

The following example enables BGP4 cooperative route filtering.

```
device# configure terminal
device(config)# rbridge-id 122
device(config-rbridge-id-122)# ip prefix-list Routesfrom10234 deny 10.20.0.0/24
device(config-rbridge-id-122)# ip prefix-list Routesfrom10234 permit 10.0.0.0/0 le 32
device(config-rbridge-id-122)# router bgp
device(config-bgp-router)# address-family ipv4 unicast
device(config-bgp-ipv4u)# neighbor 10.2.3.4 prefix-list Routesfrom1234 in
device(config-bgp-ipv4u)# neighbor 10.2.3.4 capability orf prefixlist send
```

BGP4 confederations

A large autonomous system (AS) can be divided into multiple subautonomous systems and grouped into a single BGP4 confederation.

Each subautonomous system must be uniquely identified within the confederation AS by a subautonomous system number. Within each subautonomous system, all the rules of internal BGP (iBGP) apply. For example, all BGP routers inside the subautonomous system must be fully meshed. Although eBGP is used between subautonomous systems, the subautonomous systems within the confederation exchange routing information like iBGP peers. Next hop, Multi Exit Discriminator (MED), and local preference information is preserved when crossing subautonomous system boundaries. To the outside world, a confederation looks like a single AS.

The AS path list is a loop-avoidance mechanism used to detect routing updates leaving one subautonomous system and attempting to re-enter the same subautonomous system. A routing update attempting to re-enter a subautonomous system it originated from is detected because the subautonomous system sees its own subautonomous system number listed in the update's AS path.

Configuring BGP4 confederations

BGP4 confederations, composed of multiple subautonomous systems, can be created.

1. Enter the **configure terminal** command to access global configuration mode.

```
device# configure terminal
```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

```
device(config)# rbridge-id 122
```

3. Enter the **router bgp** command to enable BGP routing.

```
device(config-rbridge-id-122)# router bgp
```

4. Enter the **local-as** command to configure the autonomous system number (ASN) in which your device resides.

```
device(config-bgp-router)# local-as 65520
```

5. Enter the **confederation identifier** command and specify an ASN to configure a BGP confederation identifier.

```
device(config-bgp-router)# confederation identifier 100
```

6. Enter the **confederation peers** command and specify as many ASNs as needed to list all BGP peers that will belong to the confederation.

```
device(config-bgp-router)# confederation peers 65520 65521 65522
```

The following example creates a confederation with the confederation ID "100" and adds three subautonomous systems to the confederation.

```
device# configure terminal
device(config)# rbridge-id 122
device(config-rbridge-id-122)# router bgp
device(config-bgp-router)# local-as 65520
device(config-bgp-router)# confederation identifier 100
device(config-bgp-router)# confederation peers 65520 65521 65522
```

BGP community and extended community

A BGP community is a group of destinations that share a common property. Community information identifying community members is included as a path attribute in BGP UPDATE messages. You can perform actions on a group using community and extended community attributes to trigger routing decisions.

All communities of a particular type can be filtered out, or certain values can be specified for a particular type of community. You can also specify whether a particular community is transitive or non-transitive across an autonomous system (AS) boundary.

An extended community is an 8-octet value and provides a larger range for grouping or categorizing communities. BGP extended community attributes are specified in RFC 4360.

You define the extended community list using the **ip extcommunity-list** command. The extended community can then be matched or applied to the neighbor through the route map. The route map must be applied on the neighbor to which routes need to carry the extended community attributes. The "send-community" should be enabled for the neighbor configuration to start including the attributes while sending updates to the neighbor.

Defining BGP4 extended communities

In order to apply a BGP4 extended community filter, a BGP4 extended community filter must be defined.

1. Enter the **configure terminal** command to access global configuration mode.

```
device# configure terminal
```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

```
device(config)# rbridge-id 122
```

3. Enter the **ip extcommunity-list** command and specify a number to set a BGP extended community filter.

```
device(config-rbridge-id-122)# ip extcommunity-list 1 permit soo 123:2
```

4. Enter the **route-map** *name* command to create and define a route map and enter route-map configuration mode.

```
device(config-rbridge-id-122)# route-map extComRmap permit 10
```

   Permits a matching pattern.

5. Enter the **match extcommunity** command and specify an extended community list number.

```
device(config-route-map-extComRmap/permit/10)# match extcommunity 1
```

6. Enter the **exit** command.

```
device(config-route-map-extComRmap/permit/10)# exit
```

7. Enter the **route-map***name* command to define a route map and enter route-map configuration mode.

```
device(config-rbridge-id-122)# route-map sendExtComRmap permit 10
```

   Permits a matching pattern.

8. Enter the **set extcommunity** command and specify the **rt** *extcommunity value* keyword to specify the route target (RT) extended community attribute.

```
device(config-route-map-sendExtComRmap/permit/10)# set extcommunity rt 3:3
```

9. Enter the **set extcommunity** command and specify the **soo** *extcommunity value* keyword to the site of origin (SOO) extended community attribute.

```
device(config-route-map-sendExtComRmap/permit/10)# set extcommunity soo 2:2
```

The following example configures an extended community ACL called “extended”, defines a route map, and permits and sets a matching pattern.

```
device# configure terminal
device(config)# rbridge-id 122
device(config-rbridge-id-122)# ip extcommunity-list 1 permit soo 123:2
device(config-rbridge-id-122)# route-map extComRmap permit 10
device(config-route-map-extComRmap/permit/10)# match extcommunity 1
device(config-route-map-extComRmap/permit/1u)# exit
device(config-rbridge-id-122)# route-map sendExtComRmap permit 10
device(config-route-map-sendExtComRmap/permit/10)# set extcommunity rt 3:3
device(config-route-map-sendExtComRmap/permit/10)# set extcommunity soo 2:2
```

Applying a BGP4 extended community filter

A BGP4 extended community filter can be applied.

BGP4 communities must already be defined.

1. Enter the **configure terminal** command to access global configuration mode.

```
device# configure terminal
```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

```
device(config)# rbridge-id 122
```

3. Enter the **ip extcommunity-list** command and specify a number to set a BGP extended community filter.

```
device(config-rbridge-id-122)# ip extcommunity-list 1 permit rt 123:2
```

4. Enter the **router bgp** command to enable BGP routing.

```
device(config-rbridge-id-122)# router bgp
```

5. Enter the **local-as** command to configure the autonomous system number (ASN) in which your device resides.

```
device(config-bgp-router)# local-as 1000
```

6. Enter the **neighbor** *ip-address* **remote-as** command to specify the ASN in which the remote neighbor resides.

```
device(config-bgp-router)# neighbor 10.1.2.3 remote-as 1001
```

7. Enter the **address-family ipv4 unicast** command to enter IPv4 address family configuration mode.

```
device(config-bgp-router)# address-family ipv4 unicast
```

8. Enter the **neighbor** *ip-address* **activate** command to enable the exchange of information with the neighbor.

```
device(config-bgp-ipv4u)# neighbor 10.1.2.3 activate
```

9. Enter the **neighbor** *ip-address* **route-map** command and specify the **in** keyword to apply a route map to incoming routes.

```
device(config-bgp-ipv4u)# neighbor 10.1.2.3 route-map in extComRmapt
```

10. Enter the **neighbor** *ip-address* **route-map** command and specify the **out** keyword to apply a route map to outgoing routes.

```
device(config-bgp-ipv4u)# neighbor 10.1.2.3 route-map out sendExtComRmap
```

11. Enter the **neighbor** *ip-address* **send-community** command and specify the **both** keyword to enable the sending of standard and extended attributes in updates to the specified BGP neighbor.

```
device(config-bgp-ipv4u)# neighbor 10.1.2.3 send-community both
```

The following example applies a BGP4 extended community filter.

```
device# configure terminal
device(config)# rbridge-id 122
device(config-rbridge-id-122)# ip extcommunity-list 1 permit rt 123:2
device(config-rbridge-id-122)# router bgp
device(config-bgp-router)# local-as 1000
device(config-bgp-router)# neighbor 10.1.2.3 remote-as 1001
device(config-bgp-router)# address-family ipv4 unicast
device(config-bgp-ipv4u)# neighbor 10.1.2.3 activate
device(config-bgp-ipv4u)# neighbor 10.1.2.3 route-map in extComRmapt
device(config-bgp-ipv4u)# neighbor 10.1.2.3 route-map out sendExtComRmap
device(config-bgp-ipv4u)# neighbor 10.1.2.31 send-community both
```

BGP4 graceful restart

BGP4 graceful restart (GR) allows for restarts where BGP neighboring devices participate in the restart, helping to ensure that no route and topology changes occur in the network for the duration of the restart.

The GR feature provides a routing device with the capability to inform its neighbors when it is performing a restart.

When a BGP session is established, GR capability for BGP is negotiated by neighbors through the BGP OPEN message. If the neighbor also advertises support for GR, GR is activated for that neighbor session. If neither peer exchanges the GR capability, the session is not GR-capable. If the BGP session is lost, the BGP peer router, known as a GR helper, marks all routes associated with the device as “stale” but continues to forward packets to these routes for a set period of time. The restarting device also continues to forward packets for the

duration of the graceful restart. When the graceful restart is complete, routes are obtained from the helper so that the device is able to quickly resume full operation.

When the GR feature is configured on a device, both helper router and restarting router functionalities are supported. It is not possible to disable helper functionality explicitly.

GR is disabled by default and can be enabled for both IPv4 and IPv6 address families. When the GR timer expires, the BGP RASlog message is triggered.

NOTE
BGP4 GR can be configured for a global routing instance or for a specified VRF instance.

Configuring BGP4 graceful restart

The graceful restart (GR) feature can be configured on a routing device, providing it with the capability to inform its neighbors when it is performing a restart.

NOTE
High availability (HA) requires GR to be enabled.

1. Enter the **configure terminal** command to access global configuration mode.

```
device# configure terminal
```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

```
device(config)# rbridge-id 122
```

3. Enter the **router bgp** command to enable BGP routing.

```
device(config-rbridge-id-122)# router bgp
```

4. Enter the **local-as** command to configure the autonomous system number (ASN) in which your device resides.

```
device(config-bgp-router)# local-as 1000
```

5. Enter the **neighbor** *ip address* **remote-as** command to add a neighbor.

```
device(config-bgp-router)# neighbor 10.11.12.13 remote-as 2
```

6. Enter the **address-family ipv4 unicast** command to enter IPv4 address-family configuration mode.

```
device(config-bgp-router)# address-family ipv4 unicast
```

7. Enter the **graceful-restart** command to enable the graceful restart feature.

```
device(config-bgp-ipv4u)# graceful-restart
```

8. Do any of the following:
   - Enter the **graceful-restart** command and use **purge-time** parameter to overwrite the default purge-time value.

     ```
     device(config-bgp-ipv4u)# graceful-restart purge-time 300
     ```

   - Enter the **graceful-restart** command and use **restart-time** parameter to overwrite the default restart-time advertised to graceful restart-capable neighbors.

     ```
     device(config-bgp-ipv4u)# graceful-restart restart-time 180
     ```

   - Enter the **graceful-restart** command and use **stale-routes-time** parameter to overwrite the default amount of time that a helper device will wait for an EOR message from a peer.

     ```
     device(config-bgp-ipv4u)# graceful-restart stale-routes-time 100
     ```

The following example enables the GR feature.

```
device# configure terminal
device(config)# rbridge-id 122
device(config-rbridge-id-122)# router bgp
device(config-bgp-router)# local-as 1
device(config-bgp-router)# neighbor 10.11.12.13 remote-as 2
device(config-bgp-router)# address-family ipv4 unicast
device(config-bgp-ipv4u)# graceful-restart
```

The following example enables the GR feature and sets the purge time to 300 seconds, over-writing the default value.

```
device# configure terminal
device(config)# rbridge-id 122
device(config-rbridge-id-122)# router bgp
device(config-bgp-router)# local-as 1
device(config-bgp-router)# neighbor 10.11.12.13 remote-as 2
device(config-bgp-router)# address-family ipv4 unicast
device(config-bgp-ipv4u)# graceful-restart purge-time 180
```

The following example enables the GR feature and sets the restart time to 180 seconds, over-writing the default value.

```
device# configure terminal
device(config)# rbridge-id 122
device(config-rbridge-id-122)# router bgp
device(config-bgp-router)# local-as 1
device(config-bgp-router)# neighbor 10.11.12.13 remote-as 2
device(config-bgp-router)# address-family ipv4 unicast
device(config-bgp-ipv4u)# graceful-restart restart-time 180
```

The following example enables the GR feature and sets the stale-routes time to 100 seconds, over-writing the default value.

```
device# configure terminal
device(config)# rbridge-id 122
device(config-rbridge-id-122)# router bgp
device(config-bgp-router)# local-as 1
device(config-bgp-router)# neighbor 10.11.12.13 remote-as 2
device(config-bgp-router)# address-family ipv4 unicast
device(config-bgp-ipv4u)# graceful-restart stale-routes-time 100
```

Use the **clear ip bgp neighbor** command with the **all** parameter for the changes to the GR parameters to take effect immediately.

BGP dynamic neighbors

The BGP dynamic neighbors feature allows peering to a group of remote neighbors defined by a listen range. BGP neighbors can be created without statically configuring them.

Routing protocols are designed to exchange routing information and distribute it to all neighbors. Each protocol has its own way of detecting neighbors and establishing adjacency with them. By design, all IGP neighbors are a single hop away. However, this is not true for BGP. BGP neighbors can be a single hop away or *n* hops away. This implies that BGP neighbors are not dynamically discovered. Rather, an administrator must enable and configure each neighbor for the exchange of routing information. The greater the number of neighbors, the greater the administrative overhead. With BGP dynamic neighbors, when a device sees an incoming TCP session initiated from a remote neighbor with an IP address in the configured subnet listen range, a new BGP neighbor is dynamically created. The device that initiates this connection still has the neighbor statically configured. Each listen range can be configured as a subnet IP range. BGP dynamic neighbors are configured using a range of IP addresses and BGP peer groups.

Whenever the incoming TCP session falls under the configured subnet listen range, a new BGP neighbor is dynamically created as a member of that peer group. Once the listen range is configured, and the associated peer group is activated, dynamic BGP neighbor creation does not require any further configuration on the initial device. Newly created BGP dynamic peers automatically inherit the attributes associated with the peer group attached to the listen range, such as inbound policy, outbound policy, and so on.

You can set the total number of dynamic neighbors that can be formed in the system. When the global or listen range limit is increased, any new connection coming from the remote end that falls under the configured range is accepted. If the limit has already been reached and you reduce the global or peer-group limit, already established dynamic neighbors are not destroyed. Once the limit has been reached and new connection requests are received, the connections are rejected and the information is logged.

BGP dynamic neighbors are deleted when a session moves from the established state to the idle state.

NOTE
BGP dynamic neighbors supports only IPv4 BGP peering.

Consider the following when using BGP dynamic neighbors:

- A neighbor can be associated with a peer group so that the peer group properties are inherited by each individual neighbor.
- Individual dynamic neighbors cannot be configured with a separate set of policies as BGP dynamic neighbors are created on demand.
- BFD for dynamic neighbors is supported. When Bidirectional Forwarding Detection (BFD) detects that a link has gone down, the dynamic neighbor moves back to the idle state and the dynamic neighbor is deleted.
- Static neighbors are always given precedence over BGP dynamic neighbors.
- The **auto-shutdown-new neighbors** command is not supported for BGP dynamic neighbors.
- BTSH and MD5 passwords are not supported for BGP dynamic neighbors.

Graceful restart for BGP dynamic neighbors

Consider the following when configuring graceful restart (GR) for dynamic neighbors.

- A dynamic neighbor negotiates the graceful restart (GR) capability with its neighbors in the same way as a statically configured neighbor. Dynamic neighbors are re-established after a warm reboot. The newly formed neighbors follow the same restart protocol as that of statically configured neighbors. Dynamically created neighbors initiate the TCP connect after a GR. However, as is not the case with static neighbors, if the dynamic neighbors do not come up within the configured restart time they are deleted. If a dynamic neighbor does come up, but fails to receive the EOR within the configured purge time, all the received routes are deleted. However, the dynamic neighbors are not deleted.

- When a dynamic neighbor moves from the established state to the idle state, all the routes are cleaned up and the neighbor is deleted. Deleting the neighbor breaks the existing GR protocol. If a dynamic neighbor has negotiated the GR and the session is moving to idle, the neighbor is deleted.

Configuring BGP dynamic neighbors

BGP dynamic neighbors can be configured. The following task sets a global limit on the number of dynamic BGP neighbors, creates a BGP peer group. and associates a subnet range with the peer group. The peer group is added to the BGP neighbor table of the local device and an alternate ASN is configured. For information on configuring BGP dynamic neighbors for an IP fabric, refer to the *Network OS IP Fabrics Configuration Guide*.

1. Enter the **configure terminal** command to access global configuration mode.

```
device# configure terminal
```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

```
device(config)# rbridge-id 122
```

3. Enter the **router bgp** command to enable BGP routing.

```
device(config-rbridge-id-122)# router bgp
```

4. Enter the **local-as** command to configure the autonomous system number (ASN) in which your device resides.

```
device(config-bgp-router)# local-as 65520
```

5. Enter the **neighbor peer-group** command, specifying a name for the peer group, to create a peer-group.

```
device(config-bgp-router)# neighbor mypeergroup1 peer-group
```

6. Enter the **neighbor** *peer-group* **remote-as** command to specify an AS for the peer-group.

```
device(config-bgp-router)# neighbor mypeergroup1 remote-as 80
```

7. Enter the **listen-range** command, specifying an IP address and mask, to associate a subnet range with the BGP peer group. Use the **limit** parameter and specify a value to set the maximum number of BGP dynamic neighbors that can be created.

```
device(config-bgp-router)# listen-range 150.1.0.0/16 peer-group mypeergroup1 limit 20
```

8. Enter the **neighbor alternate-as** command with the **add** parameter, specifying a value, to set an alternate AS number for listen range neighbors in the configured peer group.

```
device(config-bgp-router)# neighbor mypeergroup1 alternate-as add 101
```

The following example sets a global limit on the number of dynamic BGP neighbors, creates a BGP peer group, and associates a subnet range with the peer group. The peer group is added to the BGP neighbor table of the local device and an alternate ASN is configured.

```
device# configure terminal
device(config)# rbridge-id 122
device(config-rbridge-id-122)# router bgp
device(config-bgp-router)# local-as 65520
device(config-bgp-router)# neighbor mypeergroup1 peer-group
device(config-bgp-router)# neighbor mypeergroup1 remote-as 80
device(config-bgp-router)# listen-range 150.1.0.0/16 peer-group mypeergroup1 limit 20
device(config-bgp-router)# neighbor mypeergroup1 alternate-as add 101
```

BGP add path overview

The BGP add path feature provides the ability for multiple paths for the same prefix to be advertised without the new paths implicitly replacing the previous paths. Path diversity is achieved rather than path hiding.

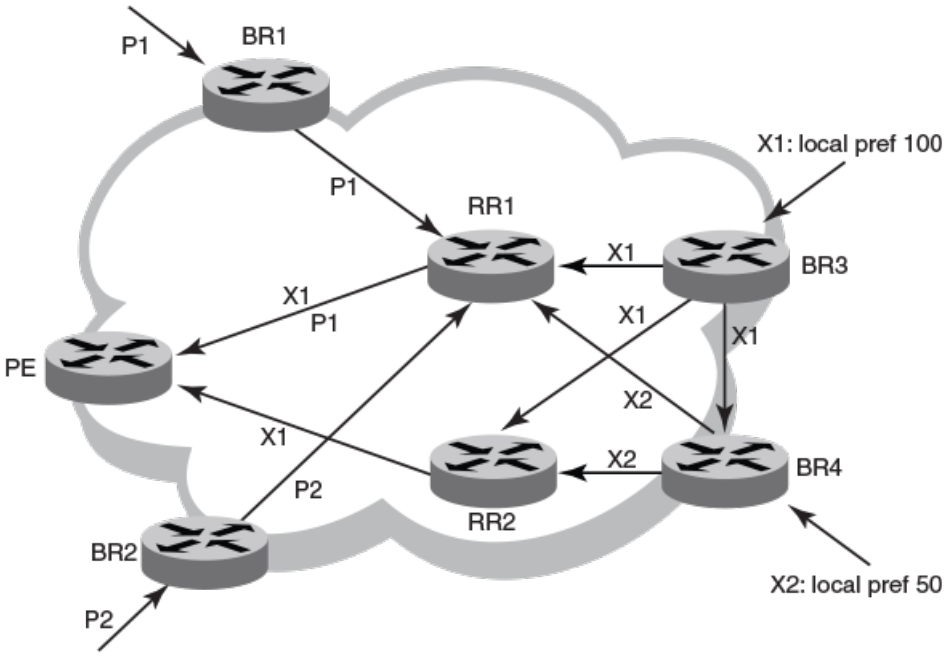
BGP devices and route reflectors (RRs) generally advertise only their best path to neighboring devices, even when multiple paths exist. The advertisement of the same prefix from the same neighbor replaces the previous announcement of that prefix. This is known as an implicit withdraw, behavior that achieves better scaling but at the cost of path diversity. When RRs are deployed inside an autonomous system (AS) to avoid iBGP scaling issues, only the best route is reflected even if multiple Equal-Cost Multipath (ECMP) routes exist. These secondary best paths remain hidden.

Path hiding can affect the efficient use of BGP multipath and path diversity, and prevent hitless planned maintenance. Upon next hop failures, path hiding also inhibits fast and local recovery because the network has to wait for BGP control plane convergence to restore traffic. BGP add path enables BGP to advertise even the secondary best routes so that multiple paths for the same prefix can be advertised without the new paths implicitly replacing previous paths. BGP add path provides a generic way of offering path diversity.

In the figure below, path hiding occurs in two ways:

- Prefix P has paths P1 and P2 advertised from BR1 and BR2 to RR1. RR1 selects P1 as the best path and advertises only P1 to PE.
- Prefix X has path X1 advertised from BR3 to BR4 with local preference 100. BR4 also has path X2. However, only the best path, X1, is selected. BR3 advertises X1 to the RRs and X2 is suppressed.

FIGURE 24 BGP path hiding



NOTE
BGP add path is supported for both route reflectors and confederations.

NOTE
BGP add path is supported for both VCS fabrics and IP fabrics, but is supported for the IPv4 and IPv6 unicast address families only.

Advantages and limitations of BGP add path

The following are the advantages and limitations provided by BGP add path.

Advantages of BGP add path

- **Fast convergence and fault tolerance**: When BGP add path is enabled, more than one path to a destination is advertised. If one of the paths goes down, connectivity is easily restored due to the availability of backup paths. If the next hop for the prefix becomes unreachable, the device can switch to the backup route immediately without having to wait for BGP control plane messages.
- **Enhanced load balancing capabilities**: Traditionally with RRs in an iBGP domain, only the best path is given to the clients even if ECMP paths exists. This affects load balancing. With additional paths advertised by RRs, the clients have more effective load balance.

Limitations of BGP add path

- Load balancing is achieved for the BGP IPv4 and IPv6 address family neighbors only and not for EVPN neighbors. For more information on BGP EVPN, refer to the BGP EVPN chapter.
- BGP add path does not provide any advantage for ARP and MAC routes supported by EVPN peers. L2 ECMP is not currently supported.
- BGP multipath must be configured to choose ECMP paths. Only the best path and the first 5 ECMP paths are chosen as additional paths for basic functionality support and advertised to neighbors with path IDs.

BGP add path functionality

BGP add path is implemented by including an additional four octet value known as a path identifier (ID) for each path in the NLRI. Path IDs are unique to a peering session and are generated for each network.

A path ID can apply to the IPv4 or IPv6 unicast address family.

Therefore, when the same prefix is received with the same path ID from the same peer, it is considered as a replacement route or a duplicate route. When the same prefix is received with a different path ID from the same peer, it is considered as an additional path to the prefix.

To send or receive additional paths, the add path capability must be negotiated. If it isn't negotiated, only the best path can be sent. BGP updates carry the path ID once the add path capability is negotiated. In order to carry the path ID in an update message, the existing NLRI encodings are extended by prepending the path ID field, which consists of four octets.

The assignment of the path ID for a path by a BGP device occurs in such a way that the BGP device is able to use the prefix and path ID to uniquely identify a path advertised to a neighbor so as to continue to send further updates for that path. The receiving BGP neighbor that re-advertises a route regenerates its own path ID to be associated with the re-advertised route.

The set of additional paths advertised to each neighbor can be different, and advertise filters are provided on a per neighbor basis.

Negotiating BGP4 add paths capability

BGP4 neighbors can be configured to send or receive additional paths after negotiation is completed.

1. Enter the **configure terminal** command to access global configuration mode.

```
device# configure terminal
```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

```
device(config)# rbridge-id 122
```

3. Enter the **router bgp** command to enable BGP routing.

```
device(config-rbridge-id-122)# router bgp
```

4. Enter the **local-as** command to configure the autonomous system number (ASN) in which your device resides.

```
device(config-bgp-router)# local-as 65520
```

5. Enter the **neighbor** *ipv4 address* **remote-as** command to add a neighbor.

```
device(config-bgp-router)# neighbor 10.10.10.1 remote-as 2
```

6. Enter the **address-family unicast** command using the **ipv4** parameter to enter BGP address-family IPv4 unicast configuration mode.

```
device(config-bgp-router)# address-family ipv4 unicast
```

7. Enter the **neighbor capability additional-paths** command, specifying an IP address, to enable the advertisement of additional paths. Enter the **send** parameter to enable BGP4 to send additional paths to BGP neighbors.

```
device(config-bgp-ipv4u)# neighbor 10.10.10.1 capability additional-paths send
```

8. Enter the **neighbor capability additional-paths** command, specifying an IP address, to enable the advertisement of additional paths. Enter the **receive** parameter to enable BGP4 toreceive additional paths from BGP neighbors.

```
device(config-bgp-ipv4u)# neighbor 10.10.10.1 capability additional-paths receive
```

The following example enables BGP4 to receive additional paths from BGP neighbors and to send additional paths to BGP neighbors.

```
device# configure terminal
device(config)# rbridge-id 122
device(config-rbridge-id-122)# router bgp
device(config-bgp-router)# local-as 65520
device(config-bgp-router)# neighbor 10.10.10.1 remote-as 2
device(config-bgp-router)# address-family ipv4 unicast
device(config-bgp-ipv4u)# neighbor 10.10.10.1 capability additional-paths send
device(config-bgp-ipv4u)# neighbor 10.10.10.1 capability additional-paths receive
```

Advertising best BGP4 additional paths

Additional paths that the device selects as best paths can be advertised.

1. Enter the **configure terminal** command to access global configuration mode.

```
device# configure terminal
```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

```
device(config)# rbridge-id 122
```

3. Enter the **router bgp** command to enable BGP routing.

```
device(config-rbridge-id-122)# router bgp
```

4. Enter the **local-as** command to configure the autonomous system number (ASN) in which your device resides.

```
device(config-bgp-router)# local-as 65520
```

5. Enter the **neighbor** *ipv4 address* **remote-as** command to add a neighbor.

```
device(config-bgp-router)# neighbor 10.10.10.1 remote-as 2
```

6. Enter the **address-family unicast** command using the **ipv4** parameter to enter BGP address-family IPv4 unicast configuration mode.

```
device(config-bgp-router)# address-family ipv4 unicast
```

7. Enter the **neighbor additional-paths advertise** command, specifying an IP address, using the **best** *value* parameter to configure BGP4 to advertise the best BGP path and an additional four routes to this neighbor.

```
device(config-bgp-ipv4u)# neighbor 10.10.10.1 additional-paths advertise best 4
```

The following example configures BGP4 to advertise the best BGP path and an additional four routes a to a BGP neighbor.

```
device# configure terminal
device(config)# rbridge-id 122
device(config-rbridge-id-122)# router bgp
device(config-bgp-router)# local-as 65520
device(config-bgp-router)# neighbor 10.10.10.1 remote-as 2
device(config-bgp-router)# address-family ipv4 unicast
device(config-bgp-router)# neighbor 10.10.10.1 additional-paths advertise best 4
```

Advertising all BGP4 additional paths

Additional paths can be advertised.

1. Enter the **configure terminal** command to access global configuration mode.

```
device# configure terminal
```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

```
device(config)# rbridge-id 122
```

3. Enter the **router bgp** command to enable BGP routing.

```
device(config-rbridge-id-122)# router bgp
```

4. Enter the **local-as** command to configure the autonomous system number (ASN) in which your device resides.

```
device(config-bgp-router)# local-as 65520
```

5. Enter the **neighbor** *ipv4 address* **remote-as** command to add a neighbor.

```
device(config-bgp-router)# neighbor 10.10.10.1 remote-as 2
```

6. Enter the **address-family unicast** command using the **ipv4** parameter to enter BGP address-family IPv4 unicast configuration mode.

```
device(config-bgp-router)# address-family ipv4 unicast
```

7. Enter the **neighbor additional-paths advertise** command, specifying an IP address, using the **all** parameter to configure BGP4 to advertise all BGP4 paths with a unique next hop.

```
device(config-bgp-ipv4u)# neighbor 10.10.10.1 additional-paths advertise all
```

The following example configures BGP4 to advertise all BGP additional paths.

```
device# configure terminal
device(config)# rbridge-id 122
device(config-rbridge-id-122)# router bgp
device(config-bgp-router)# local-as 65520
device(config-bgp-router)# neighbor 10.10.10.1 remote-as 2
device(config-bgp-router)# address-family ipv4 unicast
device(config-bgp-ipv4u)# neighbor 10.10.10.1 additional-paths advertise all
```

Auto shutdown of BGP neighbors on initial configuration

The auto shutdown of BGP neighbors on initial configuration feature prevents a BGP peer from attempting to establish connections with remote peers when the BGP peer is first configured. Sessions are only initiated when the entire configuration for the BGP peer is complete.

When the auto shutdown of BGP neighbors on initial configuration feature is enabled, the value of the shutdown parameter for any existing configured neighbor is not changed. Any new BGP neighbor configured after the feature is enabled has the shutdown state set to the configured value. When the feature is enabled and a new BGP neighbor is configured, the shutdown parameter of the BGP neighbor is automatically set to enabled. When all the BGP neighbor parameters are configured and it is ready to start the establishment of BGP session with the remote peer, the shutdown parameter of the BGP neighbor is then disabled.

Any new neighbor configured and added to a peer group has the shutdown flag enabled by default. Additionally, any new neighbor configured with the autonomous system (AS) in which that remote neighbor resides specified using the **neighbor remote-as** command has the shutdown flag enabled by default.

Configuring auto shutdown of BGP neighbors on initial configuration

Follow this procedure to enable auto shutdown of BGP neighbors on initial configuration.

1. Enter the **configure terminal** command to access global configuration mode.

```
device# configure terminal
```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

```
device(config)# rbridge-id 122
```

3. Enter the **router bgp** command to enable BGP routing.

```
device(config-rbridge-id-122)# router bgp
```

4. Enter the **local-as** command to configure the autonomous system number (ASN) in which your device resides.

```
device(config-bgp-router)# local-as 65520
```

5. Enter the **auto-shutdown-new-neighbors** command to prevent the BGP peer from attempting to establish connections with remote peers until all BGP neighbor parameters are configured.

```
device(config-bgp-router)# auto-shutdown-new-neighbors
```

The following example enables auto shutdown of BGP neighbors on initial configuration.

```
device# configure terminal
device(config)# rbridge-id 122
device(config-rbridge-id-122)# router bgp
device(config-bgp-router)# local-as 65520
device(config-bgp-router)# auto-shutdown-new-neighbors
```

Disabling the BGP4 peer shutdown state

When the auto shutdown of BGP4 neighbors on initial configuration feature is enabled, a BGP4 peer is prevented from attempting to establish connections with remote peers when the BGP4 peer is first configured. Once all of the configuration parameters for the peer are complete, you can start the BGP4 session establishment process and disable the peer shutdown state. Follow this procedure to disable the peer shutdown state.

1. Enter the **configure terminal** command to access global configuration mode.

```
device# configure terminal
```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

```
device(config)# rbridge-id 122
```

3. Enter the **router bgp** command to enable BGP routing.

```
device(config-rbridge-id-122)# router bgp
```

4. Enter the **local-as** command to configure the autonomous system number (ASN) in which your device resides.

```
device(config-bgp-router)# local-as 65520
```

5. Enter the **no neighbor shutdown** command, specifying an IP address, to disable the peer shutdown state and begin the BGP4 session establishment process.

```
device(config-bgp-router)# no neighbor 10.1.1.1 shutdown
```

The following example disables the peer shutdown state and begins the BGP4 session establishment process.

```
device# configure terminal
device(config)# rbridge-id 122
device(config-rbridge-id-122)# router bgp
device(config-bgp-router)# local-as 65520
device(config-bgp-router)# no neighbor 10.1.1.1 shutdown
```

BGP4 graceful shutdown

BGP4 graceful shutdown automatically reroutes traffic to an alternate path prior to shutting down a neighbor, minimizing traffic loss during a planned shutdown and reestablishment of BGP4 sessions.

When a planned maintenance is carried out, traffic loss is generally expected until BGP converges in the network. BGP graceful shutdown provides benefits, such as fewer lost packets and less time spent reconfiguring devices by reducing the loss of inbound or outbound traffic flows forwarded along a peering link that is being shut down for planned maintenance.

BGP graceful shutdown helps to mitigate traffic loss so that traffic is automatically rerouted while the neighbor is shutdown.

When BGP graceful shutdown is initiated on one or more BGP neighbors, BGP starts a graceful shutdown timer on the graceful shutdown initiator and then performs a route refresh to all graceful shutdown neighbors with updated BGP attributes, such as LOCAL_PREFRENCE or confederations for iBGP, or AS_PATH prepend for eBGP. The best path calculation on all the graceful shutdown is then triggered, making the routes via the graceful shutdown initiator less preferable. Thus, the traffic from graceful shutdown neighbors is re-routed to alternate paths. However, the path via the graceful shutdown initiator is still valid until the graceful shutdown timer expires. Once the graceful shutdown timer expires, all the graceful shutdown neighbors are shut down on the graceful shutdown initiator and all traffic is re-routed to alternate paths. The operator can now carry the maintenance operation.

For the graceful shutdown initiator, if there is any outbound policy associated with a particular graceful shutdown neighbor, the outbound policy is applied before the graceful shutdown parameters are applied.

> NOTE
> For the graceful shutdown neighbor, if there is any inbound policy associated with the graceful shutdown initiator modifying the LOCAL_PREFERENCE or ASPATH, the graceful shutdown parameters become insignificant. In order to prevent this, it is recommended to modify the existing inbound policy rule to match the graceful shutdown community attribute and make the route less preferred. Therefore, the send-community attribute should also be negotiated between graceful shutdown initiator and neighbor.

Configuring graceful shutdown for all BGP4 neighbors

The graceful shutdown feature can be configured on a routing device so that traffic is automatically rerouted to an alternate path prior to shutting down a BGP4 neighbor, minimizing traffic loss during a planned shutdown and reestablishment of BGP4 sessions. A specific BGP4 neighbor or all BGP4 neighbors can be gracefully shut down. The following task gracefully shuts down all BGP4 neighbors.

1. Enter the **configure terminal** command to access global configuration mode.

```
device# configure terminal
```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

```
device(config)# rbridge-id 122
```

3. Enter the **router bgp** command to enable BGP routing.

```
device(config-rbridge-id-122)# router bgp
```

4. Enter the **local-as** command to configure the autonomous system number (ASN) in which your device resides.

```
device(config-bgp-router)# local-as 1000
```

5. Enter the **neighbor** *ip address* **remote-as** command to add a neighbor.

```
device(config-bgp-router)# neighbor 10.11.12.13 remote-as 2
```

6. Enter the **graceful-shutdown** command with the **community** and **local-preference** parameters to gracefully shut down all BGP4 neighbors, set the graceful shutdown time, and set the community and local preference attributes.

```
device(config-bgp-router)# graceful-shutdown 600 community 10
```

The following example gracefully shuts down all BGP neighbors and sets the graceful shutdown timer to 180 seconds. The route map "myroutemap" is specified for graceful shutdown attributes.

```
device# configure terminal
device(config)# rbridge-id 122
device(config-rbridge-id-122)# router bgp
device(config-bgp-router)# local-as 1000
device(config-bgp-router)# neighbor 10.11.12.13 remote-as 2
device(config-bgp-router)# graceful-shutdown 600 community 10
```

Configuring graceful shutdown for a BGP4 peer group

The graceful shutdown feature can be configured on a routing device so that traffic is automatically rerouted to an alternate path prior to shutting down a BGP4 peer group, minimizing traffic loss during a planned shutdown and reestablishment of BGP4 sessions. The following task gracefully shuts down a BGP4 peer group.

1. Enter the **configure terminal** command to access global configuration mode.

```
device# configure terminal
```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

```
device(config)# rbridge-id 122
```

3. Enter the **router bgp** command to enable BGP routing.

```
device(config-rbridge-id-122)# router bgp
```

4. Enter the **local-as** command to configure the autonomous system number (ASN) in which your device resides.

```
device(config-bgp-router)# local-as 1000
```

5. Enter the **neighbor** *peer-group-name* **peer-group** command to add a peer group.

```
device(config-bgp-router)# neighbor grp-1 peer-group
```

6. Enter the **neighbor** *peer-group* **graceful-shutdown** command and specify a value to gracefully shut down the BGP4 peer group and set the graceful shutdown timer. Use the **community** parameter to set the community attribute.

```
device(config-bgp-router)# neighbor grp-1 graceful-shutdown 600 community 20
```

The following example gracefully shuts down a BGP4 peer group and sets the graceful shutdown timer to 600 seconds. The community attribute is set to 20

```
device# configure terminal
device(config)# rbridge-id 122
device(config-rbridge-id-122)# router bgp
device(config-bgp-router)# local-as 1000
device(config-bgp-router)# neighbor grp-1 peer-group
device(config-bgp-router)# neighbor grp-1 graceful-shutdown 600 community 20
```

Configuring graceful shutdown for a specified BGP4 neighbor

The graceful shutdown feature can be configured on a routing device so that traffic is automatically rerouted to an alternate path prior to shutting down a BGP4 neighbor, minimizing traffic loss during a planned shutdown and reestablishment of BGP4 sessions. A specific BGP4 neighbor or all BGP4 neighbors can be gracefully shut down. The following task gracefully shuts down a specified BGP4 neighbor.

1. Enter the **configure terminal** command to access global configuration mode.

```
device# configure terminal
```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

```
device(config)# rbridge-id 122
```

3. Enter the **router bgp** command to enable BGP routing.

```
device(config-rbridge-id-122)# router bgp
```

4. Enter the **local-as** command to configure the autonomous system number (ASN) in which your device resides.

```
device(config-bgp-router)# local-as 1000
```

5. Enter the **neighbor** *ip address* **remote-as** command to add a neighbor.

```
device(config-bgp-router)# neighbor 10.11.12.13 remote-as 2
```

6. Enter the **neighbor** *ip address* **graceful-shutdown** command and specify a value to gracefully shut down the BGP4 neighbor and set the graceful shutdown timer. Use the **community** parameter to set the community attribute.

```
device(config-bgp-router)# neighbor 10.11.22.23 graceful-shutdown 600 commuity 30
```

The following example gracefully shuts down a specified BGP4 neighbor and sets the graceful shutdown timer to 600 seconds. The community attribute is set to 30.

```
device# configure terminal
device(config)# rbridge-id 122
device(config-rbridge-id-122)# router bgp
device(config-bgp-router)# local-as 1000
device(config-bgp-router)# neighbor 10.11.12.13 remote-as 2
device(config-bgp-router)# neighbor 10.11.22.23 graceful-shutdown 600 commuity 30
```

BGP automatic neighbor discovery

BGP automatic neighbor discovery is primarily used for automatically bringing BGP up in the underlay network of an IP Fabric, significantly reducing the configuration involved. When BGP automatic neighbor discovery is configured, individual neighbors and remote AS configurations do not need to be configured. Only the local AS and automatic neighbor discovery need to be configured on one of the BGP peer groups.

Therefore, for an IP Fabric, if a new leaf or spine node is added to the topology, only the local AS and the peer group automatic discovery configurations need to be configured on the newly added node. No further configuration changes are required on the existing nodes of the IP Fabric. The situation where the neighbor remote AS needs to be configured for each spine node and on each spine node, and for each of the leaf nodes, is avoided.

NOTE
For security purposes, a peer group must have an MD5 password. The MD5 password created should be the same across all nodes.

NOTE
The primary use for BGP automatic neighbor discovery is for automatically bringing BGP up in the underlay network of the IP Fabric topology. For more information refer to *Extreme Network OS IP Fabrics Configuration Guide*.

BGP automatic neighbor discovery is achieved using the new LLDP Type, Length, and Values (TLV) with the BGP information. When BGP automatic neighbor discovery is enabled, each node plays the role of both sender and receiver.

The role of sender

- BGP informs LLDP whenever the local AS is configured, modified or removed, and the **neighbor accept-lldp-neighbors** command is configured for one of the peer groups and an MD5 password set.
- Upon receiving the local AS information from BGP, LLDP broadcasts the information across all the router ports on the device using the new LLDP TLV.
- LLDP also encodes the IP address of the respective router port, or the donor IP in the case of un-numbered interfaces in the new TLV.
- LLDP maintains this information and periodically advertises it.
- When an interface which is configured as a router port comes up or goes down, LLDP updates and sends the TLV accordingly.

The role of receiver

- When LLDP receives the new TLV with the BGP local AS and the IP Address information from the peer, it stores the information and forwards it to BGP.
- BGP then creates a peer with the received IP address as the neighbor IP and the received local AS numbers as the neighbor remote AS. Once the peer is created, the TCP session is initiated.
- The same process is repeated on all the nodes in the IP Fabric, automatically bringing up BGP in the Underlay.

BGP automatic neighbor discovery limitations

The following limitations apply when configuring BGP4 automatic neighbor discovery.

BGP automatic neighbor discovery is:

- Supported only for IPv4.
- Supported only on router ports.
- Supported only for the default VRF.
- Supported for Layer 3 PO and unnumbered interfaces.
- Not supported for BGP multihop sessions.
- Not supported for BGP Confederations.

In addition:

- BGP4 automatic neighbor discovery works only between supported Extreme devices. Interoperability with other vendors is not possible.

NOTE
BGP automatic neighbor discovery is primarily used for automatically bringing BGP up in the underlay network of the IP Fabric Topology. For more information refer to the *Extreme Network OS IP Fabrics Configuration Guide*.

Static and automatic neighbor peer transition

- If a static neighbor is initially configured and the same peer is then automatically discovered through LLDP, static neighbor configuration takes precedence over the dynamic peer. The peer continues to stay static as along as the configuration exists. Once the static configuration is removed, the dynamic peer is created automatically.
- If a static neighbor is configured and the automatically discovered peer already exists, then the dynamic peer is reset and a new TCP session is initiated with the static configuration. If the static neighbor configuration is removed, then the dynamic neighbor is created automatically. Once the static configuration is removed, the dynamic peer is created.

Enabling BGP automatic neighbor discovery

BGP automatic neighbor discovery can be configured on a routing device so that only the local AS and automatic neighbor discovery need to be configured on one of the BGP peer groups in an IP fabric. The following task enables BGP automatic neighbor discovery for a peer group and configures an MD5 password for the peer group.

NOTE
For security purposes, a peer group must have an MD5 password. The MD5 password created should be the same across all nodes.

1. Enter the **configure terminal** command to access global configuration mode.

```
device# configure terminal
```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

```
device(config)# rbridge-id 122
```

3. Enter the **router bgp** command to enable BGP routing.

```
device(config-rbridge-id-122)# router bgp
```

4. Enter the **local-as** command to configure the autonomous system number (ASN) in which your device resides.

```
device(config-bgp-router)# local-as 100
```

5. Enter the **neighbor** *peer-group-name* **peer-group** command to add a peer group.

```
device(config-bgp-router)# neighbor lldp-grp peer-group
```

6. Enter the **neighbor** *peer-group-name* **accept-lldp-neighbors** command to enable BGP automatic neighbor discovery for the peer group.

```
device(config-bgp-router)# neighbor lldp-grp accept-lldp-neighbors
```

7. Enter the **neighbor** *peer-group-name* **password** command to specify a MD5 password for the peer group.

```
device(config-bgp-router)# neighbor lldp-grp password mysecurepassword
```

The following example enables BGP automatic neighbor discovery for a peer group and specifies that the peer group number is 1. An MD5 password is also configured for the peer-group.

```
device# configure terminal
device(config)# rbridge-id 122
device(config-rbridge-id-122)# router bgp
device(config-bgp-router)# local-as 100
device(config-bgp-router)# neighbor lldp-grp peer-group
device(config-bgp-router)# neighbor lldp-grp accept-lldp-neighbors
device(config-bgp-router)# neighbor lldp-grp password mysecurepassword
```

Disabling the advertisement of TLV values for automatically discovered BGP neighbors

Link Layer Discovery Protocol (LLDP) is can be configured to disable the advertisement of the Type, Length, and Values (TLV) values for automatically discovered BGP neighbors. The advertisement of the TLV values for automatically discovered BGP neighbors is enabled by default as long the feature is configured through BGP. The following task disables the advertising of BGP neighbor TLV in LLDP.

1. Enter the **configure terminal** command to access global configuration mode.

   LLDP can be configured to disable advertisements of bgp-auto-nbr-tlvs either in global LLDP mode or per interface.

   ```
   device# configure terminal
   ```

2. Enter the **protocol lldp** command to enter LLDP configuration mode.

   ```
   device(config)# protocol lldp
   ```

3. Enter the **no advertise bgp-auto-nbr-tlv** command to enable the advertising of BGP neighbor TLV in LLDP.

   ```
   device(conf-lldp)# advertise bgp-auto-nbr-tlv
   ```

The following example disables the advertising of BGP neighbor TLV in LLDP.

```
device# configure terminal
device(config)# protocol lldp
device(conf-lldp)# no advertise bgp-auto-nbr-tlv
```

Enabling the advertising of BGP neighbor TLV at interface level

Link Layer Discovery Protocol (LLDP) can be configured to advertise the Type, Length, and Values (TLV) values for automatically discovered BGP neighbors. The following task enables the advertising of BGP neighbor TLV for a specific LLDP profile for a specified 40-gigabit Ethernet interface.

NOTE
IF LLDP is enabled at the global level, it is automatically enabled on all interfaces. Therefore, this configuration is not mandatory and is only applicable where LLDP is not enabled at the global level.

1. Enter the **configure terminal** command to access global configuration mode.

   ```
   device# configure terminal
   ```

2. Enter the **protocol lldp** command to enter LLDP configuration mode.

   ```
   device(config)# protocol lldp
   ```

3. Enter the **profile** command and specify a name to create a LLDP profile.

```
device(conf-lldp)# profile test1
```

4. Enter the **advertise bgp-auto-nbr-tlv** command to enable the advertising of BGP neighbor TLV for the configured LLDP profile.

```
device(conf-profile-test1)# advertise bgp-auto-nbr-tlv
```

5. Enter the **exit** command until you reach global configuration mode.

```
device(conf-profile-test1)# exit
device(conf-lldp)# exit
```

6. Enter the **interface** command, specifying an interface, to access interface configuration mode.

```
device(config)# interface fortygigabitethernet 1/0/49
```

7. Enter the **lldp profile** command to apply the configured LLDP profile to the interface.

```
device (conf-if-fo-1/0/49)# lldp profile 1
```

The following example enables the advertising of BGP neighbor TLV for a specific LLDP profile for a specified 40-gigabit Ethernet interface.

```
device# configure terminal
device(config)# protocol lldp
device(conf-lldp)# profile test1
device(conf-profile-test1)# advertise bgp-auto-nbr-tlv
device(conf-profile-test1)# exit
device(conf-lldp)# exit
device(config)# interface fortygigabitethernet 1/0/49
device (conf-if-fo-1/0/49)# lldp profile 1
```

Generalized TTL Security Mechanism support

Generalized TTL Security Mechanism (GTSM) is a lightweight security mechanism that protects external Border Gateway Protocol (eBGP) peering sessions from CPU utilization-based attacks using forged IP packets. GTSM prevents attempts to hijack the eBGP peering session by a host on a network segment that is not part of either BGP network, or by a host on a network segment that is not between the eBGP peers.

GTSM is enabled by configuring a minimum Time To Live (TTL) value for incoming IP packets received from a specific eBGP peer. BGP establishes and maintains the session only if the TTL value in the IP packet header is equal to or greater than the TTL value configured for the peering session. If the value is less than the configured value, the packet is silently discarded and no Internet Control Message Protocol (ICMP) message is generated.

In the case of directly connected neighbors, the device expects the BGP control packets received from the neighbor to have a TTL value of either 254 or 255. For multihop peers, the device expects the TTL for BGP control packets received from the neighbor to be greater than or equal to 255, minus the configured number of hops to the neighbor. If the BGP control packets received from the neighbor do not have the expected value, the device drops them.

Assumptions and limitations

- GTSM is supported for both directly connected peering sessions and multihop eBGP peering sessions.
- GTSM is supported for eBGP only.

- GTSM does not protect the integrity of data sent between eBGP peers and does not validate eBGP peers through any authentication method.
- GTSM validates only the locally configured TTL count against the TTL field in the IP packet header.
- GTSM should be configured on each participating device to maximize the effectiveness of this feature.
- When GTSM is enabled, the eBGP session is secured in the incoming direction only and has no effect on outgoing IP packets or the remote device.

Configuring GTSM for BGP4

Generalized TTL Security Mechanism (GTSM) can be configured to protect external Border Gateway Protocol (eBGP) peering sessions.

1. Enter the **configure terminal** command to access global configuration mode.

```
device# configure terminal
```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

```
device(config)# rbridge-id 122
```

3. Enter the **router bgp** command to enable BGP routing.

```
device(config-rbridge-id-122)# router bgp
```

4. Enter the **local-as** command to configure the autonomous system number (ASN) in which your device resides.

```
device(config-bgp-router)# local-as 65520
```

5. Enter the **neighbor remote-as** command to add a neighbor.

```
device(config-bgp-router)# neighbor 10.10.10.1 remote-as 2
```

6. Enter the **neighbor ebgp-btsh** command, specifying an IP address, to enable GTSM.

```
device(config-bgp-router)# neighbor 10.10.10.1 ebgp-btsh
```

The following example enables GTSM between a device and a neighbor with the IP address 10.10.10.1.

```
device# configure terminal
device(config)# rbridge-id 122
device(config-rbridge-id-122)# router bgp
device(config-bgp-router)# local-as 65520
device(config-bgp-router)# neighbor 10.10.10.1 remote-as 2
device(config-bgp-router)# neighbor 10.10.10.1 ebgp-btsh
```

Disabling the BGP AS_PATH check function

A device can be configured so that the AS_PATH check function for routes learned from a specific location is disabled, and routes that contain the recipient BGP speaker's AS number are not rejected.

1. Enter the **configure terminal** command to access global configuration mode.

```
device# configure terminal
```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

```
device(config)# rbridge-id 122
```

3. Enter the **router bgp** command to enable BGP routing.

```
device(config-rbridge-id-122)# router bgp
```

4. Enter the **address-family ipv4 unicast** command to enter BGP IPv4 address family unicast configuration mode.

```
device(config-bgp-router)# address-family ipv4 unicast
```

5. Enter the **neighbor allowas-in** command and specify a **number** to disable the BGP AS_PATH check function, and specify the number of times that the AS path of a received route may contain the recipient BGP speaker's AS number and still be accepted.

```
device(config-bgp-ipv6u)# neighbor 10.1.1.1 allowas-in 3
```

The following example specifies that the AS path of a received route may contain the recipient BGP speaker's AS number three times and still be accepted.

```
device# configure terminal
device(config)# rbridge-id 122
device(config-rbridge-id-122)# router bgp
device(config-bgp-router)# address-family ipv4 unicast
device(config-bgp-ipv4u)# neighbor 10.1.1.1 allowas-in 3
```

Using route maps

A route map is a named set of match conditions and parameter settings that the device can use to modify route attributes and to control redistribution of the routes into other protocols. A route map consists of a sequence of instances, the equivalent of rows in a table. The device evaluates a route according to route map instances in ascending numerical order. The route is first compared against instance 1, then against instance 2, and so on. When a match is found, the device stops evaluating the route.

Route maps can contain **match** clauses and **set** statements. Each route map contains a **permit** or **deny** statement for routes that match the **match** clauses:

- If the route map contains a **permit** statement, a route that matches a match statement is permitted; otherwise, the route is denied.
- If the route map contains a **deny** statement, a route that matches a match statement is denied.
- If a route does not match any **match** statements in the route map, then the route is denied. This is the default action. To change the default action, configure the last **match** statement in the last instance of the route map to **permit any any** .
- If there is no **match** statement, the software considers the route to be a match.
- For route maps that contain address filters, AS-path filters, or community filters, if the action specified by a filter conflicts with the action specified by the route map, the route map action takes precedence over the filter action.

If the route map contains **set** statements, routes that are permitted by the route map **match** statements are modified according to the **set** statements.

Match statements

Match statements compare the route against one or more of the following:

- The route BGP4 MED (metric)
- The IP address of the next hop device
- The route tag
- For OSPF routes only, the route type (internal, external type 1, or external type 2)
- An AS-path access control list (ACL)
- A community ACL

- An IP prefix list
- An extended-community-based ACL

Set statements

For routes that match all of the **match** statements, for all the routes that are permitted by the route-map match statements, the route map **set** statements can perform one or more of the following modifications to the route attributes:

- Prepend AS numbers to the front of the route AS-path. By adding AS numbers to the AS-path, you can cause the route to be less preferred when compared to other routes based on the length of the AS-path.
- Add a user-defined tag or an automatically calculated tag to the route.
- Set the community attributes.
- Set the extended community attributes.
- Set the local preference.
- Set the MED (metric).
- Set the IP address of the next-hop device.
- Set the origin to IGP or INCOMPLETE.
- Set the weight.

When you configure parameters for redistributing routes into BGP4, one of the optional parameters is a route map. If you specify a route map as one of the redistribution parameters, the device matches the route against the match statements in the route map. If a match is found and if the route map contains **set** statements, the device sets the attributes in the route according to the set statements.

Refer to the following commands in the *Extreme Network OS Command Reference* for more information on creating and using route maps:

- continue
- match (route-map)
- route-map
- set as-path prepend

Matching on an AS-path

A route map that matches on a specified AS-path access control list (ACL) can be configured.

An AS-path ACL must be configured using the **ip as-path access-list** command.

1. Enter the **configure terminal** command to access global configuration mode.

```
device# configure terminal
```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

```
device(config)# rbridge-id 5
```

3. Enter the **route-map** command using the **permit** parameter and specifying a route map name to create a route map instance and allow a matching pattern.

```
device(config-rbridge-id-5)# route-map myaclroutemap1 permit 10
```

4. Enter the **match** command and specify an ACL name to configure the route map to match on the specified AS-path ACL.

```
device(config-route-map-myaclroutemap1/permit/10)# match as-path myaspath
```

The following example configures a route map instance that matches on AS-path ACL “myaspath”.

```
device# configure terminal
device(config)# rbridge-id 5
device(config-rbridge-id-5)# route-map myaclroutemap1 permit 10
device(config-route-map-myaclroutemap1/permit/10)# match as-path myaspath
```

Matching on a community ACL

A route map instance that matches on a specified community access control list (ACL) can be configured.

1. Enter the **configure terminal** command to access global configuration mode.

```
device# configure terminal
```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

```
device(config)# rbridge-id 5
```

3. Enter the **ip community-list standard** command, specifying a community number and using the **permit** parameter to create a community ACL that permits traffic.

```
device(config-rbridge-id-5)# ip community-list standard 1 permit 123:2
```

4. Enter the **route-map** command using the **permit** parameter and specifying a route map name to create a route map instance and allow a matching pattern.

```
device(config-rbridge-id-5)# route-map mycommroutemap1 permit 10
```

5. Enter the **match community** command, and specify a community number to match the BGP community access list name in the configured route-map instance.

```
device(config-route-map-mycommroutemap1/permit/10)# match community 1
```

The following example matches community ACL “1” for route map instance “mycommroutemap1”.

```
device# configure terminal
device(config)# rbridge-id 5
device(config-rbridge-id-5)# ip community-list standard 1 permit 123:2
device(config-rbridge-id-5)# route-map mycommroutemap1 permit 10
device(config-route-map-mycommroutemap1/permit/10)# match community 1
```

Matching on a destination network

A route map that matches on a destination network can be configured.

1. Enter the **configure terminal** command to access global configuration mode.

```
device# configure terminal
```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

```
device(config)# rbridge-id 5
```

3. Enter the **route-map** command using the **permit** parameter and specifying a route map name to create a route map instance and allow a matching pattern.

```
device(config-rbridge-id-5)# route-map mynetroutemap1 permit 10
```

4. Enter the **match** command with the **ip address** parameter. Specify a prefix list using the **ip prefix-list** *string* parameter to configure the route map to match on the specified prefix.

```
device(config-route-map-mynetroutemap1/permit/10)# match ip address prefix-list mylist
```

The following example configures a route map instance that matches on a specified destination network.

```
device# configure terminal
device(config)# rbridge-id 5
device(config-rbridge-id-5)# route-map mynetroutemap1 permit 10
device(config-route-map-mynetroutemap1/permit/10)# match ip address prefix-list mylist
```

Matching on a BGP4 static network

A route map that matches on a static network can be configured. In this task, a route-map is configured to match on the BGP4 static network. The device is then configured to filter the outgoing route updates to a specified BGP neighbor according to the set of attributes defined in the configured route map.

1. Enter the **configure terminal** command to access global configuration mode.

```
device# configure terminal
```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

```
device(config)# rbridge-id 5
```

3. Enter the **route-map** command using the **permit** parameter and specifying a route map name to create a route map instance and allow a matching pattern.

```
device(config-rbridge-id-5)# route-map mystaticroutemap3 permit 1
```

4. Enter the **match** command with the **protocol bgp static-network** parameter to match on BGP4 static network routes.

```
device(config-routemap-mystaticroutemap3/permit/1)# match protocol bgp static-network
```

5. Enter the **set local-preference** command and enter a value to set a BGP local-preference path attribute in the route-map instance.

```
device(config-routemap-mystaticroutemap3/permit/1)# set local-preference 150
```

6. Enter the **set community** command with the **no export** parameter to set the BGP community attribute for the route-map instance not to export to the next AS.

```
device(config-routemap-mystaticroutemap3/permit/1)# set community no-export
```

7. Enter the **exit** command to exit route map configuration mode.

```
device(config-routemap-mystaticroutemap3/permit/1)# exit
```

8. Enter the **router bgp** command to enable BGP routing.

```
device(config-rbridge-id-122)# router bgp
```

9. Enter the **address-family ipv4 unicast** command to enter IPv4 address family configuration mode.

```
device(config-bgp-router)# address-family ipv4 unicast
```

10. Enter the **neighbor** *ip-address* **route-map** command and specify the **out** keyword, specifying the route map name, to filter the outgoing route updates to a specified BGP neighbor according to the set of attributes defined in the configured route map.

```
device (config-bgp-ipv4u)# neighbor 10.1.2.3 route-map out mystaticroutemap3
```

The following example configures a route map instance that matches on a BGP4 static network and configures the device to filter the outgoing route updates to a specified BGP neighbor according to the set of attributes defined in the configured route map.

```
device# configure terminal
device(config)# rbridge-id 5
device(config-rbridge-id-5)# route-map mystaticroutemap3 permit 1
device(config-routemap-mystaticroutemap3/permit/1)# match protocol bgp static-network
device(config-routemap-mystaticroutemap3/permit/1)# set local-preference 150
device(config-routemap-mystaticroutemap3/permit/1)# set community no-export
device(config-routemap-mystaticroutemap3/permit/1)# exit
device(config-rbridge-id-5)# router bgp
device(config-bgp)# address-family ipv4 unicast
device(config-bgp-ipv4u)# neighbor 10.1.2.3 route-map out mystaticroutemap3
```

Matching on a next-hop device

A route map that matches on a next-hop device can be configured.

A prefix list must be configured using the **ip prefix-list** command.

1. Enter the **configure terminal** command to access global configuration mode.

```
device# configure terminal
```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

```
device(config)# rbridge-id 5
```

3. Enter the **route-map** command using the **permit** parameter and specifying a route map name to create a route map instance and allow a matching pattern.

```
device(config-rbridge-id-5)# route-map myhoproutemap1 permit 10
```

4. Enter the **match** command, using the **next-hop** parameter and specify a prefix-list, to match IP next-hop match conditions for a specified prefix list in a route-map instance.

```
device(config-route-map-myhoproutemap1/permit/10)# match ip next-hop    prefix-list mylist
```

The following example configures a route map and specifies a prefix list to match on a next-hop device.

```
device# configure terminal
device(config)# rbridge-id-5
device(config-rbridge-id-5)# route-map myhoproutemap1 permit 10
device(config-route-map-myhoproutemap1/permit/10)# match ip next-hop    prefix-list mylist
```

Matching on an interface

To configure a route map that matches on an interface:

```
device# configure terminal
device(config)# rbridge-id 5
device(config-rbridge-id-5)# route-map myintroutemap1 permit 99
device(config-route-map-myintroutemap1/permit/99)# match interface tengigabitethernet 5/1/1 loopback 1
```

Using route-map continue statements

Continuation statements can be configured in a route map. This task configures a route map instance and adds two route-map continue statements to the route map instance.

1. Enter the **configure terminal** command to access global configuration mode.

   ```
   device# configure terminal
   ```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

   ```
   device(config)# rbridge-id 5
   ```

3. Enter the **route-map** command using the **permit** parameter and specifying a route map name to create a route map instance and allow a matching pattern.

   ```
   device(config-rbridge-id-5)# route-map mcontroutemap1 permit 1
   ```

4. Enter the **match** command with the metric parameter and specify a value to match a route metric in the route-map instance.

   ```
   device(config-route-map-mcontroutemap1/permit/1)# match metric 10
   ```

5. Enter the **set weight** command and specify a value to set a BGP weight for the routing table in the route-map instance.

   ```
   device(config-route-map-mcontroutemap1/permit/1)# set weight 10
   ```

6. Enter the **continue** command and specify a value to configure a route-map instance number that goes in a continue statement in the route-map instance.

   ```
   device(config-routemap-mycontroutemap/permit/1)# continue 2
   ```

7. Enter the **exit** command to exit route map configuration mode.

   ```
   device(config-routemap-mycontroutemap/permit/1)# exit
   ```

8. Enter the **route-map** command using the **permit** parameter and specifying a route map name to create a route map instance and allow a matching pattern.

   ```
   device(config-rbridge-id-5)# route-map mcontroutemap1 permit 2
   ```

9. Enter the **match** command with the metric parameter and specify a value to match a route metric in the route-map instance.

   ```
   device(config-route-map-mcontroutemap1/permit/2)# match metric 10
   ```

10. Enter the **set weight** command and specify a value to set a BGP weight for the routing table in the route-map instance.

    ```
    device(config-route-map-mcontroutemap1/permit/2)# set weight 20
    ```

11. Enter the **continue** command and specify a value to configure a route-map instance number that goes in a continue statement in the route-map instance.

```
device(config-routemap-mycontroutemap/permit/2)# continue 3
```

The following example configures a route map instance and adds two route-map continue statements to the route map instance.

```
device# configure terminal
device(config)# rbridge-id 5
device(config-rbridge-id-5)# route-map mcontroutemap1 permit 1
device(config-routemap-mcontroutemap1/permit/1)# match metric 10
device(config-routemap-mcontroutemap1/permit/1)# set weight 10
device(config-routemap-mycontroutemap1/permit/1)# match metric 10
device(config-routemap-mycontroutemap1/permit/1)# continue 2
device(config-routemap-mycontroutemap1/permit/1)# exit
device(config-rbridge-id-5)# route-map mcontroutemap1 permit 2
device(config-routemap-mycontroutemap1/permit/2)# match tag 10
device(config-routemap-mycontroutemap1/permit/2)# set weight 20
```

Route-map continue statement for BGP4 routes

A continue statement in a route-map directs program flow to skip over route-map instances to another, user-specified instance. If a matched instance contains a continue statement, the system looks for the instance that is identified in the statement.

The continue statement in a matching instance initiates another traversal at the instance specified. The system records all of the matched instances and, if no deny statements are encountered, proceeds to execute the set clauses of the matched instances.

If the system scans all route-map instances but finds no matches, or if a deny condition is encountered, then it does not update the routes. Whenever a matched instance contains a deny statement, the current traversal terminates, and none of the updates specified in the set statements of the matched instances in both current and previous traversals are applied to the routes.

This supports a more programmable route-map configuration and route filtering scheme for BGP4 peering. It can also execute additional instances in a route map after an instance is executed by means of successful match statements. You can configure and organize more-modular policy definitions to reduce the number of instances that are repeated within the same route map.

This feature currently applies to BGP4 routes only. For protocols other than BGP4, continue statements are ignored.

Using a route map to configure dampening

You can set a BGP route-flap dampening penalty in a route-map instance..

1. Enter the **configure terminal** command to access global configuration mode.

```
device# configure terminal
```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

```
device(config)# rbridge-id 5
```

3. Enter the **route-map** command using the **permit** parameter and specifying a route map name to create a route map instance and allow a matching pattern.

```
device(config-rbridge-id-5)# route-map myroutemap permit 1
```

4. Enter the **set dampening** command and specify a value name to set the BGP route-flap dampening penalty for the route-map instance.

```
device(config-route-map-myroutemap/permit/1)# set dampening 20
```

The following example configures a route map instance and sets a BGP route-flap dampening penalty of 20.

```
device# configure terminal
device(config)# rbridge-id 5
device(config-rbridge-id-5)# route-map myroutemap permit 1
device(config-route-map-myroutemap/permit/1)# set dampening 20
```

Clearing diagnostic buffers

The device stores the following BGP4 diagnostic information in buffers:

- The first 400 bytes of the last packet received that contained an error
- The last NOTIFICATION message either sent or received by the device

This information can be useful if you are working with Extreme Technical Support to resolve a problem. The buffers do not identify the system time when the data was written to the buffer. If you want to ensure that diagnostic data in a buffer is recent, you can clear the buffers. You can clear the buffers for a specific neighbor or for all neighbors.

If you clear the buffer containing the first 400 bytes of the last packet that contained errors, all the bytes are changed to zeros. The Last Connection Reset Reason field of the BGP4 neighbor table also is cleared.

If you clear the buffer containing the last NOTIFICATION message sent or received, the buffer contains no data.

You can clear the buffers for all neighbors, for an individual neighbor, or for all the neighbors within a specific peer group.

Refer to the *Extreme Network OS Command Reference* for more information.

Displaying BGP4 statistics

Various **show ip bgp** commands verify information about BGP4 configurations.

Use one or more of the following commands to verify BGP4 information. The commands do not have to be entered in this order.

1. Enter the **show ip bgp summary** command.

```
device# show ip bgp summary

BGP4 Summary
Router ID: 10.10.1.4   Local AS Number: 200
Confederation Identifier: not configured
Confederation Peers:
Maximum Number of IP ECMP Paths Supported for Load Sharing: 1
Number of Neighbors Configured: 67, UP: 67
Number of Routes Installed: 258088, Uses 22195568 bytes
Number of Routes Advertising to All Neighbors: 17,035844(3,099146 entries), Uses 192,147052 bytes
Number of Attribute Entries Installed: 612223, Uses 55100070 bytes
Neighbor Address  AS#         State     Time     Rt:Accepted Filtered Sent ToSend
10.1.1.1          100         CONN      1h 3m16s    0          0        0    0
10.1.1.2          100         CONN      1h 3m16s    0          0        0    0
10.1.1.3          100         CONN      1h 3m16s    0          0        0    0
10.1.1.4          100         CONN      1h 3m16s    0          0        0    0
10.1.1.5          100         CONN      1h 3m16s    0          0        122  0
10.1.1.6          100         CONN      1h 3m16s    0          0        122  0
10.1.1.7          100         CONN      1h 3m16s    0          0        0    2
```

   This example output gives summarized BGP4 information.

2. Enter the **show ip bgp routes** command, using the **summary** keyword.

```
device# show ip bgp routes summary
  Total number of BGP routes (NLRIs) Installed     : 20
  Distinct BGP destination networks                : 20
  Filtered BGP routes for soft reconfig            : 100178
  Routes originated by this router                 : 2
  Routes selected as BEST routes                   : 19
  BEST routes not installed in IP forwarding table : 1
  Unreachable routes (no IGP route for NEXTHOP)    : 1
  IBGP routes selected as best routes              : 0
  EBGP routes selected as best routes              : 17
```

   This example shows summarized BGP4 route information.

3. Enter the **show ip bgp routes** command.

```
device# show ip bgp routes
Total number of BGP Routes: 26
Status A:AGGREGATE B:BEST b:NOT-INSTALLED-BEST C:CONFED_EBGP D:DAMPED
       E:EBGP H:HISTORY I:IBGP L:LOCAL M:MULTIPATH m:NOT-INSTALLED-MULTIPATH
       S:SUPPRESSED F:FILTERED s:STALE
      Prefix             Next Hop        MED       LocPrf     Weight Status
1     10.3.0.0/8         192.168.4.106    1         100        0      BE
         AS_PATH: 65001 4355 701 80
2     10.4.0.0/8         192.168.4.106    107       100        0      BE
         AS_PATH: 65001 4355 1
3     10.60.212.0/22     192.168.4.106    107       100        0      BE
         AS_PATH: 65001 4355 701 1 189
4     10.6.0.0/8         192.168.4.106    107       100        0      BE
         AS_PATH: 65001 4355 3356 7170 1455
5     10.8.1.0/24        192.168.4.106    0         100        0      BE
         AS_PATH: 65001
```

   This example shows general BGP4 route information.

4. Enter the **show ip bgp routes** command, using the **unreachable** keyword.

```
device# show ip bgp routes unreachable
Status A:AGGREGATE B:BEST b:NOT-INSTALLED-BEST C:CONFED_EBGP D:DAMPED
       E:EBGP H:HISTORY I:IBGP L:LOCAL M:MULTIPATH m:NOT-INSTALLED-MULTIPATH
       S:SUPPRESSED F:FILTERED s:STALE
       Prefix             Next Hop        MED     LocPrf     Weight Status
1      10.8.8.0/24        192.168.5.1     0          101        0
         AS_PATH: 65001 4355 1
```

This example shows BGP4 routes whose destinations are unreachable using any of the BGP4 paths in the BGP4 route table.

5. Enter the **show ip bgp flap-statistics** command.

```
device# show ip bgp flap-statistics
Total number of flapping routes: 414
    Status Code  >:best d:damped h:history *:valid
    Network            From         Flaps Since    Reuse    Path
h>  10.50.206.0/23   10.90.213.77   1     0 :0 :13 0 :0 :0  65001 4355 1 701
h>  10.255.192.0/20  10.90.213.77   1     0 :0 :13 0 :0 :0  65001 4355 1 7018
h>  10.252.165.0/24  10.90.213.77   1     0 :0 :13 0 :0 :0  65001 4355 1 7018
h>  10.50.208.0/23   10.90.213.77   1     0 :0 :13 0 :0 :0  65001 4355 1 701
h>  10.33.0.0/16     10.90.213.77   1     0 :0 :13 0 :0 :0  65001 4355 1 701
*>  10.17.220.0/24   10.90.213.77   1     0 :1 :4  0 :0 :0  65001 4355 701 62
```

This example shows the routes in the BGP4 route table that have changed state and have been marked as flapping routes.

BGP4+

BGP4+ overview

The implementation of IPv6 supports multiprotocol BGP (MBGP) extensions that allow Border Gateway Protocol version 4 plus (BGP4+) to distribute routing information. BGP4+ supports all of the same features and functionality as IPv4 BGP (BGP4).

IPv6 MBGP enhancements include:

- An IPv6 unicast address family and network layer reachability information (NLRI)
- Next hop attributes that use IPv6 addresses

NOTE

The 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.

BGP global mode

Configurations that are not specific to address family configuration are available in the BGP global configuration mode.

```
device(config-bgp-router)# ?

Possible completions:

  address-family                Enter Address Family command mode
  always-compare-med            Allow comparing MED from different neighbors
  as-path-ignore                Ignore AS_PATH length for best route selection
  capability                    Set capability
  cluster-id                    Configure Route-Reflector Cluster-ID
  compare-med-empty-aspath    Allow comparing MED from different neighbors even with empty as-path attribute
  compare-routerid              Compare router-id for identical BGP paths
  confederation               Configure AS confederation parameters
  default-local-preference      Configure default local preference value
  distance                      Define an administrative distance
  enforce-first-as              Enforce the first AS for EBGP routes
  fast-external-fallover        Reset session if link to EBGP peer goes down
  install-igp-cost              Install igp cost to nexthop instead of MED value as BGP cost
  local-as                      Configure local AS number
  log-dampening-debug           Log dampening debug messages
  maxas-limit                   Impose limit on number of ASes in AS-PATH attribute
  med-missing-as-worst          Consider routes missing MED attribute as least desirable
  neighbor                      Specify a neighbor router
  timers                        Adjust routing timers
```

The following neighbor configuration options are allowed under BGP global configuration mode:

```
device(config-bgp-router)# neighbor 2001:2018:8192::125 ?

Possible completions:
  advertisement-interval   Minimum interval between sending BGP routing updates
  as-override              Override matching AS-number while sending update
  bfd                      Enable BFD session for the neighbor
  capability               Advertise capability to the peer
  description              Neighbor by description
  ebgp-btsh                Enable EBGP TTL Security Hack Protection
  ebgp-multihop            Allow EBGP neighbors not on directly connected
                           networks
  enforce-first-as         Enforce the first AS for EBGP routes
  local-as                 Assign local-as number to neighbor
  maxas-limit              Impose limit on number of ASes in AS-PATH attribute
  next-hop-self            Disable the next hop calculation for this neighbor
  password                 Enable TCP-MD5 password protection
  peer-group               Create Peer Group
  remote-as                Specify a BGP neighbor
  remove-private-as        Remove private AS number from outbound updates
  shutdown                 Administratively shut down this neighbor
  soft-reconfiguration     Per neighbor soft reconfiguration
  static-network-edge      Neighbor as special service edge, static-network
                           shall not be advertised if installed as DROP
  timers                   BGP per neighbor timers
  update-source            Source of routing updates
```

IPv6 unicast address family

The IPv6 unicast address family configuration level provides access to commands that allow you to configure BGP4+ unicast routes. The commands that you enter at this level apply only to the IPv6 unicast address family.

BGP4+ supports the IPv6 address family configuration level.

You can generate a configuration for BGP4+ unicast routes that is separate and distinct from configurations for IPv4 unicast routes.

The commands that you can access while at the IPv6 unicast address family configuration level are also available at the IPv4 unicast address family configuration levels. Each address family configuration level allows you to access commands that apply to that particular address family only.

Where relevant, this chapter discusses and provides IPv6-unicast-specific examples.

The following configuration options are allowed under BGP IPv6 address family unicast mode:

```
device(config-bgp-ipv6u)# ?

Possible completions:

  aggregate-address               Configure BGP aggregate entries
  always-propagate                Allow readvertisement of best BGP routes not
                                  in IP Forwarding table
  bgp-redistribute-internal       Allow redistribution of iBGP routes into IGPs
  client-to-client-reflection     Configure client to client route reflection
  dampening                       Enable route-flap dampening
  default-information-originate   Originate Default Information
  default-metric                  Set metric of redistributed routes
  graceful-restart                Enables the BGP graceful restart capability
  maximum-paths                   Forward packets over multiple paths
  multipath                       Enable multipath for ibgp or ebgp neighbors
                                  only
  neighbor                        Specify a neighbor router
  network                         Specify a network to announce via BGP
  next-hop-enable-default         Enable default route for BGP next-hop lookup
  next-hop-recursion              Perform next-hop recursive lookup for BGP
                                  route
  redistribute                    Redistribute information from another
                                  routing protocol
  rib-route-limit                 Limit BGP rib count in routing table
  table-map                       Map external entry attributes into routing
                                  table
  update-time                     Configure igp route update interval

device(config-bgp-ipv6u)# ?

Possible completions:

  aggregate-address               Configure BGP aggregate entries
  always-propagate                Allow readvertisement of best BGP routes not
                                  in IP Forwarding table
  bgp-redistribute-internal       Allow redistribution of iBGP routes into IGPs
  client-to-client-reflection     Configure client to client route reflection
  dampening                       Enable route-flap dampening
  default-information-originate   Originate Default Information
  default-metric                  Set metric of redistributed routes
  graceful-restart                Enables the BGP graceful restart capability
  maximum-paths                   Forward packets over multiple paths
  multipath                       Enable multipath for ibgp or ebgp neighbors
                                  only
  neighbor                        Specify a neighbor router
  network                         Specify a network to announce via BGP
  next-hop-enable-default         Enable default route for BGP next-hop lookup
  next-hop-recursion              Perform next-hop recursive lookup for BGP
                                  route
  redistribute                    Redistribute information from another
                                  routing protocol
  rib-route-limit                 Limit BGP rib count in routing table
  table-map                       Map external entry attributes into routing
                                  table
  update-time                     Configure igp route update interval
```

The following neighbor configuration options are allowed under BGP IPv6 address family unicast mode:

```
device(config-bgp-ipv6u)# neighbor 2001:2018:8192::125 ?

Possible completions:
  activate                Allow exchange of route in the current family mode
```

```
additional-paths         Advertise the additional paths
allowas-in               Disables the AS_PATH check of the routes learned
                         from the AS
capability               Advertise capability to the peer
default-originate        Originate default route to peer
enable-peer-as-check     Disable routes advertise between peers in same AS
filter-list              Establish BGP filters
maximum-prefix           Maximum number of prefix accept from this peer
prefix-list              Prefix List for filtering routes
route-map                Apply route map to neighbor
route-reflector-client   Configure a neighbor as Route Reflector client
send-community           Send community attribute to this neighbor
unsuppress-map           Route-map to selectively unsuppress suppressed
                         routes
weight                   Set default weight for routes from this neighbor
```

BGP4+ neighbors

BGP4+ neighbors can be configured using link-local addresses or global addresses.

BGP4+ neighbors can be created using link-local addresses for peers in the same link. For link-local peers, the neighbor interface over which the neighbor and local device exchange prefixes is specified through the **neighbor update-source** command, and a route map is configured to set up a global next hop for packets destined for the neighbor.

To configure BGP4+ neighbors that use link-local addresses, you must do the following:

- Add the IPv6 address of a neighbor in a remote autonomous system (AS) to the BGP4+ neighbor table of the local device.
- Identify the neighbor interface over which the neighbor and local device will exchange prefixes using the **neighbor update-source** command.
- Configure a route map to set up a global next hop for packets destined for the neighbor.

The neighbor should be activated in the IPv6 address family configuration mode using the **neighbor activate** command.

BGP4+ neighbors can also be configured using a global address. The global IPv6 address of a neighbor in a remote AS must be added, and the neighbor should be activated in the IPv6 address family configuration mode using the **neighbor activate** command.

Configuring BGP4+ neighbors using global IPv6 addresses

BGP4+ neighbors can be configured using global IPv6 addresses.

1. Enter the **configure terminal** command to access global configuration mode.

```
device# configure terminal
```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

```
device(config)# rbridge-id 122
```

3. Enter the **router bgp** command to enable BGP routing.

```
device(config-rbridge-id-122)# router bgp
```

4. Enter the **local-as** command to configure the autonomous system number (ASN) in which your device resides.

```
device(config-bgp-router)# local-as 1000
```

5. Enter the **neighbor remote-as** command, specifying an IPv6 address, to specify the ASN in which the remote neighbor resides.

```
device(config-bgp-router)# neighbor 2001:db8:93e8:cc00::1 remote-as 1001
```

6. Enter the **address family ipv6 unicast** command to enter IPv6 address family configuration mode.

```
device(config-bgp-router)# address-family ipv6 unicast
```

7. Enter the **neighbor activate** command to enable the exchange of information with the neighbor.

```
device(config-bgp-ipv6u)# neighbor 2001:db8:93e8:cc00::1 activate
```

The following example configures a neighbor using a global IPv6 address.

```
device# configure terminal
device(config)# rbridge-id 122
device(config-rbridge-id-122)# router bgp
device(config-bgp-router)# local-as 1000
device(config-bgp-router)# neighbor 2001:db8:93e8:cc00::1 remote-as 1001
device(config-bgp-router)# address-family ipv6 unicast
device(config-bgp-ipv6u)# neighbor 2001:db8:93e8:cc00::1 activate
```

Configuring BGP4+ neighbors using link-local addresses

BGP4+ neighbors can be configured using link-local addresses.

1. Enter the **configure terminal** command to access global configuration mode.

```
device# configure terminal
```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

```
device(config)# rbridge-id 122
```

3. Enter the **router bgp** command to enable BGP routing.

```
device(config-rbridge-id-122)# router bgp
```

4. Enter the **local-as** command to configure the autonomous system number (ASN) in which your device resides.

```
device(config-bgp-router)# local-as 1000
```

5. Enter the **neighbor** *ipv6-address* **remote-as** command to specify the ASN in which the remote neighbor resides.

```
device(config-bgp-router)# neighbor fe80:4398:ab30:45de::1 remote-as 1001
```

6. Enter the **neighbor** *ipv6-address* **update-source** command to specify an interface.

```
device(config-bgp-router)# neighbor fe80:4398:ab30:45de::1 update-source tengigabitethernet 1/3/1
```

7. Enter the **address-family** command and specify the **ipv6** and **unicast** keywords to enter IPv6 address family configuration mode.

```
device(config-bgp-router)# address-family ipv6 unicast
```

8. Enter the **neighbor** *ipv6-address* **activate** command to enable the exchange of information with the neighbor.

```
device(config-bgp-ipv6u)# neighbor fe80:4398:ab30:45de::1 activate
```

9. Enter the **neighbor** *ipv6-address* **route-map** command and specify the **out** to apply a route map to outgoing routes.

```
device(config-bgp-ipv6u)# neighbor fe80:4398:ab30:45de::1 route-map out myroutemap
```

10. Enter the **exit** command until you return to Rbridge-ID configuration mode.

```
device(config-bgp-ipv6u)# exit
```

11. Enter the **route-map** *name* **permit** command to define the route map and enter route-map configuration mode.

```
device(config-rbridge-id-122)# route-map myroutemap permit 10
```

12. Enter the **set ipv6 next-hop** command and specify and IPv6 address to set the IPv6 address of the next hop.

```
device(config-route-map-myroutemap/permit/10)# set ipv6 next-hop 2001::10
```

The following example configures a neighbor using a link-local address and configures a route map to set up a global next hop for packets destined for the neighbor.

```
device# configure terminal
device(config)# rbridge-id 122
device(config-rbridge-id-122)# router bgp
device(config-bgp-router)# local-as 1000
device(config-bgp-router)# neighbor fe80:4398:ab30:45de::1 remote-as 1001
device(config-bgp-router)# neighbor fe80:4398:ab30:45de::1 update-source tengigabitethernet 1/3/1
device(config-bgp-router)# address-family ipv6 unicast
device(config-bgp-ipv6u)# neighbor fe80:4398:ab30:45de::1 activate
device(config-bgp-ipv6u)# neighbor fe80:4398:ab30:45de::1 route-map out myroutemap
device(config-bgp-ipv6u)# exit
device(config-bgp-router)# exit
device(config-rbridge-id-122)# route-map myroutemap permit 10
device(config-route-mapmyroutemap/permit/10)#set ipv6 next-hop 2001::10
```

BGP4+ peer groups

Neighbors having the same attributes and parameters can be grouped together by means of the **neighbor peer-group** command.

You must first create a peer group, after which you can associate neighbor IPv6 addresses with the peer group. All of the attributes that are allowed on a neighbor are allowed on a peer group as well.

BGP4+ peers and peer groups are activated in the IPv6 address family configuration mode to establish the BGP4+ peering sessions.

An attribute value configured explicitly for a neighbor takes precedence over the attribute value configured on the peer group. In the case where neither the peer group nor the individual neighbor has the attribute configured, the default value for the attribute is used.

> NOTE
> BGP4 neighbors are established and the prefixes are advertised using the **neighbor** *IP address* **remote-as** command in router BGP mode. However, when establishing BGP4+ peer sessions and exchanging IPv6 prefixes, neighbors must also be activated using the **neighbor** *IPv6 address* **activate** command in IPv6 address family configuration mode.

Configuring BGP4+ peer groups

A peer group can be created and neighbor IPv6 addresses can be associated with the peer group. The peer group is then activated in the IPv6 address family configuration mode.

1. Enter the **configure terminal** command to access global configuration mode.

```
device# configure terminal
```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

```
device(config)# rbridge-id 122
```

3. Enter the **router bgp** command to enable BGP routing.

```
device(config-rbridge-id-122)# router bgp
```

4. Enter the **local-as** command to configure the autonomous system number (ASN) in which your device resides.

```
device(config-bgp-router)# local-as 1000
```

5. Enter the **neighbor peer-group** command to create a peer group.

```
device(config-bgp-router)# neighbor mypeergroup1 peer-group
```

6. Enter the **neighbor remote-as** command, specifying a peer group, to specify the ASN of the peer group.

```
device(config-bgp-router)# neighbor mypeergroup1 remote-as 11
```

7. Enter the **neighbor peer-group** command to associate a neighbor with the peer group.

```
device(config-bgp-router)# neighbor 2001:2018:8192::125 peer-group mypeergroup1
```

8. Enter the **neighbor peer-group** command to associate another neighbor with the peer group.

```
device(config-bgp-router)# neighbor 2001:2018:8192::124 peer-group mypeergroup1
```

9. Enter the **address-family ipv6 unicast** command to enter IPv6 address family configuration mode.

```
device(config-bgp-router)# address-family ipv6 unicast
```

10. Enter the **neighbor activate** command to establish an IPv6 BGP session with the peer group.

```
device(config-bgp-ipv6u)# neighbor mypeergroup1 activate
```

The following example creates a peer group, specifying two neighbors to belong to the peer group, and activates the peer group.

```
device# configure terminal
device(config)# rbridge-id 122
device(config-rbridge-id-122)# router bgp
device(config-bgp-router)# local-as 1000
device(config-bgp-router)# neighbor mypeergroup1 peer-group
device(config-bgp-router)# neighbor mypeergroup1 remote-as 11
device(config-bgp-router)# neighbor 2001:2018:8192::125 peer-group mypeergroup1
device(config-bgp-router)# neighbor 2001:2018:8192::124 peer-group mypeergroup1
device(config-bgp-router)# address-family ipv6 unicast
device(config-bgp-ipv6u)# neighbor mypeergroup1 activate
```

Configuring a peer group with IPv4 and IPv6 peers

A peer group that contains both IPv4 and IPv6 peers can be configured.

1. Enter the **configure terminal** command to access global configuration mode.

```
device# configure terminal
```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

```
device(config)# rbridge-id 122
```

3. Enter the **router bgp** command to enable BGP routing.

```
device(config-rbridge-id-122)# router bgp
```

4. Enter the **local-as** command to configure the autonomous system number (ASN) in which your device resides.

```
device(config-bgp-router)# local-as 1000
```

5. Enter the **neighbor peer-group** command, specifying a peer group, to create a peer group.

```
device(config-bgp-router)# neighbor p1 peer-group
```

6. Enter the **neighbor remote-as** command to specify the ASN of the peer group.

```
device(config-bgp-router)# neighbor p1 remote-as 11
```

7. Enter the **neighbor peer-group** command, specifying an IPv6 address, to associate a neighbor with the peer group.

```
device(config-bgp-router)# neighbor 2001:2018:8192::124 peer-group p1
```

8. Enter the **neighbor peer-group** command, specifying a different IPv6 address, to associate another neighbor with the peer group.

```
device(config-bgp-router)# neighbor 10.0.0.1 peer-group p1
```

9. Enter the **address-family ipv6 unicast** command to enter IPv6 address family configuration mode.

```
device(config-bgp-router)# address-family ipv6 unicast
```

10. Enter the **neighbor activate** command to establish an IPv6 BGP session with the peer group.

```
device(config-bgp-ipv6u)# neighbor p1 activate
```

The following example creates a peer group with both IPv6 and IPv4 peers and activates the peer group in the IPv6 address family.

```
device# configure terminal
device(config)# rbridge-id 122
device(config-rbridge-id-122)# router bgp
device(config-bgp-router)# local-as 1000
device(config-bgp-router)# neighbor p1 peer-group
device(config-bgp-router)# neighbor p1 remote-as 11
device(config-bgp-router)# neighbor 2001:2018:8192::124 peer-group p1
device(config-bgp-router)# neighbor 10.0.0.1 peer-group p1
device(config-bgp-router)# address-family ipv6 unicast
device(config-bgp-ipv6u)# neighbor p1 activate
```

Importing routes into BGP4+

Routes can be explicitly specified for advertisement by BGP.

The routes imported into BGP4+ must first exist in the IPv6 unicast route table.

1. Enter the **configure terminal** command to access global configuration mode.

```
device# configure terminal
```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

```
device(config)# rbridge-id 122
```

3. Enter the **router bgp** command to enable BGP routing.

```
device(config-rbridge-id-122)# router bgp
```

4. Enter the **neighbor remote-as** command, specifying an IPv6 address, to specify the ASN in which the remote neighbor resides.

```
device(config-bgp-router)# neighbor fe80:4398:ab30:45de::1 remote-as 1001
```

5. Enter the **address-family ipv6 unicast** command to enter IPv6 address family configuration mode.

```
device(config-bgp-router)# address-family ipv6 unicast
```

6. Enter the **network** command and specify a *network/mask* to import the specified prefix into the BGP4+ database.

```
device(config-bgp-ipv6u)# network 2001:db8::/32
```

The following example imports the 2001:db8::/32 prefix in to the BGP4+ database for advertising.

```
device# configure terminal
device(config)# rbridge-id 122
device(config-rbridge-id-122)# router bgp
device(config-bgp-router)# neighbor fe80:4398:ab30:45de::1 remote-as 1001
device(config-bgp-router)# address-family ipv6 unicast
device(config-bgp-ipv6u)# network 2001:db8::/32
```

Advertising the default BGP4+ route

A BGP device can be configured to advertise the default IPv6 route to all BGP4+ neighbors and to install that route in the local BGP4+ route table.

The default route must be present in the local IPv6 route table.

1. Enter the **configure terminal** command to access global configuration mode.

```
device# configure terminal
```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

```
device(config)# rbridge-id 122
```

3. Enter the **router bgp** command to enable BGP routing.

```
device(config-rbridge-id-122)# router bgp
```

4. Enter the **address-family ipv6 unicast** command to enter IPv6 address family unicast configuration mode.

```
device(config-bgp-router)# address-family ipv6 unicast
```

5. Enter the **default-information-originate** command to advertise the default IPv6 route to all BGP4+ neighbors and to install that route in the local BGP4+ route table.

```
device(config-bgp-ipv6u)# default-information-originate
```

The following example enables a BGP4+ device to advertise the default IPv6 route to all BGP4+ neighbors and to install that route in the local BGP4+ route table.

```
device# configure terminal
device(config)# rbridge-id 122
device(config-rbridge-id-122)# router bgp
device(config-bgp-router)# address-family ipv6 unicast
device(config-bgp-ipv6u)# default-information-originate
```

Advertising the default BGP4+ route to a specific neighbor

A BGP device can be configured to advertise the default IPv6 route to a specific neighbor.

1. Enter the **configure terminal** command to access global configuration mode.

```
device# configure terminal
```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

```
device(config)# rbridge-id 122
```

3. Enter the **router bgp** command to enable BGP routing.

```
device(config-rbridge-id-122)# router bgp
```

4. Enter the **local-as** command to configure the autonomous system number (ASN) in which your device resides.

```
device(config-bgp-router)# local-as 1000
```

5. Enter the **address-family ipv6 unicast** command to enter IPv6 address family configuration mode.

```
device(config-bgp-router)# address-family ipv6 unicast
```

6. Enter the **neighbor default-originate** command and specify an IPv6 address to enable the BGP4+ device to advertise the default IPv6 route to a specific neighbor.

```
device(config-bgp-ipv6u)# neighbor 2001:db8:93e8:cc00::1 default-originate
```

The following example enables a BGP4+ device to advertise the default IPv6 route to a specific neighbor.

```
device# configure terminal
device(config)# rbridge-id 122
device(config-rbridge-id-122)# router bgp
device(config-bgp-router)# local-as 1000
device(config-bgp-router)# address-family ipv6 unicast
device(config-bgp-ipv6u)# neighbor 2001:db8:93e8:cc00::1 default-originate
```

Using the IPv6 default route as a valid next hop for a BGP4+ route

In certain cases, such as when a device is acting as an edge device, it can be configured to use the default route as a valid next hop.

By default, a device does not use a default route to resolve a BGP4+ next-hop route. If the IPv6 route lookup for the BGP4+ next-hop does not result in a valid IGP route (including static or direct routes), the BGP4+ next-hop is considered to be unreachable and the BGP4+ route is not used. You can configure the device to use the default route as a valid next hop.

1. Enter the **configure terminal** command to access global configuration mode.

```
device# configure terminal
```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

```
device(config)# rbridge-id 122
```

3. Enter the **router bgp** command to enable BGP routing.

```
device(config-rbridge-id-122)# router bgp
```

4. Enter the **address-family ipv6 unicast** command to enter IPv6 address family configuration mode.

```
device(config-bgp-router)# address-family ipv6 unicast
```

5. Enter the **next-hop-enable-default** command to configure the device to use the default route as a valid next hop.

```
device(config-bgp-ipv6u)# next-hop-enable-default
```

The following example configures a BGP4+ device to use the default route as a valid next hop.

```
device# configure terminal
device(config)# rbridge-id 122
device(config-rbridge-id-122)# router bgp
device(config-bgp-router)# address-family ipv6 unicast
device(config-bgp-ipv6u)# next-hop-enable-default
```

BGP4+ next hop recursion

A device can find the IGP route to the next-hop gateway for a BGP4+ route.

For each BGP4+ route learned, the device performs a route lookup to obtain the IPv6 address of the next hop for the route. A BGP4+ route is eligible for addition in the IPv6 route table only if the following conditions are true:

- The lookup succeeds in obtaining a valid next-hop IPv6 address for the route.
- The path to the next-hop IPv6 address is an IGP path or a static route path.

By default, the software performs only one lookup for the next-hop IPv6 address for the BGP4+ route. If the next hop lookup does not result in a valid next hop IPv6 address, or the path to the next hop IPv6 address is a BGP4+ path, the BGP4+ route destination is considered unreachable. The route is not eligible to be added to the IPv6 route table.

The BGP4+ route table can contain a route with a next hop IPv6 address that is not reachable through an IGP route, even though the device can reach a hop farther away through an IGP route. This can occur when the IGPs do not learn a complete set of IGP routes, so the device learns about an internal route through IBGP instead of through an IGP. In this case, the IPv6 route table will not contain a route that can be used to reach the BGP4+ route destination.

To enable the device to find the IGP route to the next-hop gateway for a BGP4+ route, enable recursive next-hop lookups. With this feature enabled, if the first lookup for a BGP4+ route results in an IBGP path that originated within the same AS, rather than an IGP path or static route path, the device performs a lookup on the next hop IPv6 address for the next hop gateway. If this second lookup results in an IGP path, the software considers the BGP4+ route to be valid and adds it to the IPv6 route table. Otherwise, the device performs another lookup on the next hop IPv6 address of the next hop for the next hop gateway, and so on, until one of the lookups results in an IGP route.

You must configure a static route or use an IGP to learn the route to the EBGP multihop peer.

Enabling next-hop recursion

Next hop recursion can be enabled so that a device can find the IGP route to the next hop gateway for a BGP4+ route.

1. Enter the **configure terminal** command to access global configuration mode.

```
device# configure terminal
```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

```
device(config)# rbridge-id 122
```

3. Enter the **router bgp** command to enable BGP routing.

```
device(config-rbridge-id-122)# router bgp
```

4. Enter the **address-family ipv6 unicast** command to enter IPv6 address family configuration mode.

```
device(config-bgp-router)# address-family ipv6 unicast
```

5. Enter the **next-hop-recursion** command to enable recursive next hop lookups.

```
device(config-bgp-ipv6u)# next-hop-recursion
```

The following example enables recursive next hop lookups.

```
device# configure terminal
device(config)# rbridge-id 122
device(config-rbridge-id-122)# router bgp
device(config-bgp-router)# address-family ipv6 unicast
device(config-bgp-ipv6u)# next-hop-recursion
```

BGP4+ NLRIs and next hop attributes

BGP4+ introduces new attributes to handle multiprotocol extensions for BGP.

Multiprotocol BGP (MBGP) is an extension to BGP that enables BGP to carry routing information for multiple address families.

BGP4+ introduces new attributes to handle multiprotocol extensions for BGP:

- Multiprotocol reachable Network Layer Reachability Information (MP_REACH_NLRI): Used to carry the set of reachable destinations, together with the next hop information, to be used for forwarding to these destinations.
- Multiprotocol unreachable NLRI (MP_UNREACH_NLRI): Used to carry the set of unreachable destinations.

MP_REACH_NLRI and MP_UNREACH_NLRI are optional and non-transitive, so that a BGP4+ speaker that does not support the multiprotocol capabilities ignores the information carried in these attributes, and does not pass it to other BGP4+ speakers. A BGP speaker that uses multiprotocol extensions for IPv6 uses the capability advertisement procedures to determine whether the speaker can use multiprotocol extensions with a particular peer.

The next hop information carried in the MP_REACH_NLRI path attribute defines the network layer address of the border router that will be used as the next hop to the destinations listed in the MP_NLRI attribute in the UPDATE message.

MP_REACH_NLRI and MP_UNREACH_NLRI carry IPv6 prefixes.

BGP4+ route reflection

A BGP device can act as a route-reflector client or as a route reflector. You can configure a BGP peer as a route-reflector client from the device that is going to reflect the routes and act as the route reflector using the **neighbor route-reflector-client** command.

When there is more than one route reflector, they should all belong to the same cluster. By default, the value for **cluster-id** is used as the device ID. The device ID can be changed using the **cluster-id** command.

The route-reflector server reflects the routes as follows:

- Routes from the client are reflected to the client as well as to nonclient peers.

- Routes from nonclient peers are reflected only to client peers.

If route-reflector clients are connected in a full IBGP mesh, you can disable client-to-client reflection on the route reflector using the **no client-to-client-reflection** command.

A BGP device advertises only those routes that are preferred ones and are installed into the Routing Table Manager (RTM). When a route cannot be installed into the RTM because the routing table is full, the route reflector may not reflect that route. In cases where the route reflector is not placed directly in the forwarding path, you can configure the route reflector to reflect routes even though those routes are not in the RTM using the **always-propagate** command.

Configuring a cluster ID for a route reflector

The cluster ID can be changed if there is more than one route reflector, so that all route reflectors belong to the same cluster.

1. Enter the **configure terminal** command to access global configuration mode.

```
device# configure terminal
```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

```
device(config)# rbridge-id 122
```

3. Enter the **router bgp** command to enable BGP routing.

```
device(config-rbridge-id-122)# router bgp
```

4. Enter the **local-as** command to configure the autonomous system number (ASN) in which your device resides.

```
device(config-bgp-router)# local-as 1000
```

5. Enter the **cluster-id** command and specify a value to change the cluster ID of a device from the default device ID.

```
device(config-bgp-router)# cluster-id 321
```

The following example changes the cluster ID of a device from the default device ID to 321.

```
device# configure terminal
device(config)# rbridge-id 122
device(config-rbridge-id-122)# router bgp
device(config-bgp-router)# cluster-id 321
```

Configuring a route reflector client

A BGP peer can be configured as a route reflector client.

1. Enter the **configure terminal** command to access global configuration mode.

```
device# configure terminal
```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

```
device(config)# rbridge-id 122
```

3. Enter the **router bgp** command to enable BGP routing.

```
device(config-rbridge-id-122)# router bgp
```

4. Enter the **local-as** command to configure the autonomous system number (ASN) in which your device resides.

```
device(config-bgp-router)# local-as 1000
```

5. Enter the **address-family ipv6 unicast** command to enter IPv6 address family configuration mode.

```
device(config-bgp-router)# address-family ipv6 unicast
```

6. Enter the **neighbor route-reflector-client** command, specifying an IPv6 address, to configure a specified neighbor to be a route reflector client.

```
device(config-bgp-ipv6u)# neighbor 2001:db8:e0ff:783a::4 route-reflector-client
```

The following example configures a neighbor with the IPv6 address 2001:db8:e0ff:783a::4 to be a route reflector client.

```
device# configure terminal
device(config)# rbridge-id 122
device(config-rbridge-id-122)# router bgp
device(config-bgp-router)# local-as 1000
device(config-bgp-router)# address-family ipv6 unicast
device(config-bgp-ipv6u)# neighbor 2001:db8:e0ff:783a::4 route-reflector-client
```

BGP4+ route aggregation

A device can be configured to aggregate routes in a range of networks into a single IPv6 prefix.

By default, a device advertises individual BGP4+ routes for all the networks. The aggregation feature allows you to configure a device to aggregate routes in a range of networks into a single IPv6 prefix. For example, without aggregation, a device will individually advertise routes for networks 2001:db8:0001:0000::/64, 2001:db8:0002:0000::/64, 2001:db8:0003:0000::/64, and so on. You can configure the device to send a single, aggregate route for the networks instead so that the aggregate route would be advertised as 2001:db8::/32 to BGP4 neighbors.

Aggregating routes advertised to BGP neighbors

A device can be configured to aggregate routes in a range of networks into a single IPv6 prefix.

The route-map should already be defined.

You can aggregate BGP4+ routes, for example 2001:db8:0001:0000::/64, 2001:db8:0002:0000::/64, 2001:db8:0003:0000::/64 into a single network prefix: 2001:db8::/24.

1. Enter the **configure terminal** command to access global configuration mode.

```
device# configure terminal
```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

```
device(config)# rbridge-id 122
```

3. Enter the **router bgp** command to enable BGP routing.

```
device(config-rbridge-id-122)# router bgp
```

4. Enter the **address-family ipv6 unicast** command to enter IPv6 address family configuration mode.

```
device(config-bgp-router)# address-family ipv6 unicast
```

5. Enter the **aggregate-address** command to aggregate the routes from a range of networks into a single network prefix.

```
device(config-bgp-ipv6u)# aggregate-address 2001:db8::/32
```

The following example enables a BGP4+ device to advertise the default route and send the default route to a specified neighbor.

```
device# configure terminal
device(config)# rbridge-id 122
device(config-rbridge-id-122)# router bgp
device(config-bgp-router)# address-family ipv6 unicast
device(config-bgp-ipv6u)# aggregate-address 2001:db8::/32
```

BGP4+ multipath

The BGP4+ multipath feature can be used to enable load-balancing across different paths.

BGP4+ selects only one best path for each IPv6 prefix it receives before installing it in the IP routing table. If you need load-balancing across different paths, you must enable BGP4+ multipath using the **maximum-paths** command under IPv6 address family configuration mode.

IBGP paths and EBGP paths can be exclusively selected, or a combination of IBGP and EBGP paths can be selected.

The following attributes of parallel paths must match for them to be considered for multipathing:

- Weight
- Local Preference
- Origin
- AS-Path Length
- MED
- Neighbor AS (EBGP multipath)
- AS-PATH match (for IBGP multipath)
- IGP metric to BGP next hop

Enabling load-balancing across different paths

The BGP4+ multipath feature can be configured, enabling load-balancing across different paths.

1. Enter the **configure terminal** command to access global configuration mode.

```
device# configure terminal
```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

```
device(config)# rbridge-id 122
```

3. Enter the **router bgp** command to enable BGP routing.

```
device(config-rbridge-id-122)# router bgp
```

4. Enter the **address-family ipv6 unicast** command to enter IPv6 address family configuration mode.

```
device(config-bgp-router)# address-family ipv6 unicast
```

5. Do one of the following:
   - Enter the **maximum-paths** command and specify a value to set the maximum number of BGP4+ shared paths.
   - Enter the **maximum-paths** command using the **use-load-sharing** keyword to set the maximum number of BGP4+ shared paths to that of the value already configured by means of the **ip load-sharing** command.

   ```
   device(config-bgp-ipv6u)# maximum-paths 8
   ```

   or

   ```
   device(config-bgp-ipv6u)# maximum-paths use-load-sharing
   ```

   NOTE
   The **ip load-sharing** command has been deprecated. Note the following use cases with respect to legacy configurations and the current release:

   1. If the **ip load-sharing** command has been used in a previous release and maximum paths **has not** been configured, then load balancing is influenced only for static routes and load balancing for the routing protocol is not affected.
   2. If the **ip load-sharing** command has been used in a previous release and maximum paths **has** been configured, then the lowest configured number of paths takes precedence.
   3. If only the **ip load-sharing** command has been used with an existing configuration and maximum paths **has not** been configured, then there is no effect on load sharing for the routing protocol.
   4. If only the **ip load-sharing** command has been used with an existing configuration and maximum paths **has** been configured, then the lowest configured number of paths takes precedence. The number of maximum paths can still be adjusted, by means of the **maximum-paths** command, prior to an upgrade to the current release.

The following example sets the maximum number of BGP4+ shared paths to 8.

```
device# configure terminal
device(config)# rbridge-id 122
device(config-rbridge-id-122)# router bgp
device(config-bgp-router)# address-family ipv6 unicast
device(config-bgp-ipv6u)# maximum-paths 8
```

The following example sets the maximum number of BGP4+ shared paths to that of the value already configured using the **ip load-sharing** command.

```
device# configure terminal
device(config)# rbridge-id 122
device(config-rbridge-id-122)# router bgp
device(config-bgp-router)# address-family ipv6 unicast
device(config-bgp-ipv6u)# maximum-paths use-load-sharing
```

Route maps

Route maps must be applied to IPv6 unicast address prefixes in IPv6 address family configuration mode.

By default, route maps that are applied under IPv4 address family configuration mode using the **neighbor route-map** command are applied to only IPv4 unicast address prefixes. To apply route maps to IPv6 unicast address prefixes, the **neighbor route-map** command must be used in IPv6 address family configuration mode. The route maps are applied as the inbound or outbound routing policy for neighbors under the specified address family. Configuring separate route maps under each address family type simplifies managing complicated or different policies for each address family.

Configuring a route map for BGP4+ prefixes

Route maps can be applied to IPv6 unicast address prefixes either as the inbound or outbound routing policy for neighbors under the specified address family.

1. Enter the **configure terminal** command to access global configuration mode.

```
device# configure terminal
```

2. Enter the **ipv6 prefix-list** command in RBridge ID configuration mode and enter a name to configure an IPv6 prefix list.

```
device(config)# rbridge-id 122
device(config-rbridge-id-122)# ipv6 prefix-list myprefixlist seq 10 permit 2001:db8::/32
```

   The prefix list name, sequence number, and permits packets are specified.

3. Enter the **route-map** *name* **permit** command to define the route map and enter route map configuration mode.

```
device(config-rbridge-id-122)# route-map myroutemap permit 10
```

4. Enter the **match ipv6 address** command and specify the name of a prefix list.

```
device(config-route-map-myroutemap/permit/10)# match ipv6 address prefix-list myprefixlist
```

5. Enter the **exit** command to return to RBridge ID configuration mode.

```
device(config-route-map-myroutemap/permit/10)# exit
```

6. Enter the **router bgp** command to enable BGP routing.

```
device(config-rbridge-id-122)# router bgp
```

7. Enter the **local-as** command to configure the autonomous system number (ASN) in which your device resides.

```
device(config-bgp-router)# local-as 1000
```

8. Enter the **neighbor** *ipv6-address* **remote-as** command to specify the ASN in which the remote neighbor resides.

```
device(config-bgp-router)# neighbor fe80:4398:ab30:45de::1 remote-as 1001
```

9. Enter the **address-family** command and specify the **ipv6** and **unicast** keywords to enter IPv6 address family configuration mode.

```
device(config-bgp-router)# address-family ipv6 unicast
```

10. Enter the **neighbor** *ipv6-address* **activate** command to enable the exchange of information with the neighbor.

```
device(config-bgp-ipv6u)# neighbor fe80:4398:ab30:45de::1 activate
```

11. Enter the **neighbor** *ipv6-address* **route-map** command and specify the **out** keyword to apply a route map to outgoing routes.

```
device(config-bgp-ipv6u)# neighbor fe80:4398:ab30:45de::1 route-map out myroutemap
```

The following example applies a route map, “myroutemap”, as the outbound routing policy for a neighbor.

```
device# configure terminal
device(config)# rbridge-id 122
device(config-rbridge-id-122)# ipv6 prefix-list myprefixlist seq 10 permit 2001:db8::/32
device(config-rbridge-id-122)# route-map myroutemap permit 10
device(config-route-map-myroutemap/permit/10)# match ipv6 address prefix-list myprefixlist
device(config-route-map-myroutemap/permit/10)# exit
device(config-rbridge-id-122)# router bgp
device(config-bgp-router)# local-as 1000
device(config-bgp-router)# neighbor fe80:4398:ab30:45de::1 remote-as 1001
device(config-bgp-router)# address-family ipv6 unicast
device(config-bgp-ipv6u)# neighbor fe80:4398:ab30:45de::1 activate
device(config-bgp-ipv6u)# neighbor fe80:4398:ab30:45de::1 route-map out myroutemap
```

Redistributing prefixes into BGP4+

Various routes can be redistributed into BGP.

Static, connected, and OSPF routes can be redistributed into BGP. This task redistributes OSPFv3 routes into BGP4+.

1. Enter the **configure terminal** command to access global configuration mode.

   ```
   device# configure terminal
   ```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

   ```
   device(config)# rbridge-id 122
   ```

3. Enter the **router bgp** command to enable BGP routing.

   ```
   device(config-rbridge-id-122)# router bgp
   ```

4. Enter the **address-family ipv6 unicast** command to enter IPv6 address family configuration mode.

   ```
   device(config-bgp-router)# address-family ipv6 unicast
   ```

5. Enter the **redistribute** command using the **ospf** and **external1** keywords to redistribute IPv6 OSPF external type 1 routes.

   ```
   device(config-bgp-ipv6u)# redistribute ospf match external1
   ```

The following example redistributes OSPF external type 1 routes into BGP4+.

```
device# configure terminal
device(config)# rbridge-id 122
device(config-rbridge-id-122)# router bgp
device(config-bgp-router)# address-family ipv6 unicast
device(config-bgp-ipv6u)# redistribute ospf match external1
```

Redistributing routes into BGP4+

Various routes can be redistributed into BGP. This task redistributes connected routes into BGP.

1. Enter the **configure terminal** command to access global configuration mode.

   ```
   device# configure terminal
   ```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

```
device(config)# rbridge-id 122
```

3. Enter the **router bgp** command to enable BGP routing.

```
device(config-rbridge-id-122)# router bgp
```

4. Enter the **address-family ipv6 unicast** command to enter IPv6 address family unicast configuration mode.

```
device(config-bgp-router)# address-family ipv6 unicast
```

5. Enter the **redistribute** command using the **connected** keyword to redistribute connected routes into BGP4+.

```
device(config-bgp-ipv6u)# redistribute connected
```

The following example redistributes connected routes into BGP4+.

```
device# configure terminal
device(config)# rbridge-id 122
device(config-rbridge-id-122)# router bgp
device(config-bgp-router)# address-family ipv6 unicast
device(config-bgp-ipv6u)# redistribute connected
```

Specifying the weight added to BGP4+ received routes

The weight that the device adds to received routes can be specified. The following task changes the weight from the default for routes that are received from a specified BGP4+ neighbor.

1. Enter the **configure terminal** command to access global configuration mode.

```
device# configure terminal
```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

```
device(config)# rbridge-id 122
```

3. Enter the **router bgp** command to enable BGP routing.

```
device(config-rbridge-id-122)# router bgp
```

4. Enter the **local-as** command to configure the autonomous system number (ASN) in which your device resides.

```
device(config-bgp-router)# local-as 65520
```

5. Enter the **address-family ipv6 unicast** command to enter address family IPv6 unicast configuration mode.

```
device(config-bgp-router)# address-family ipv6 unicast
```

6. Enter the **neighbor weight** command and specify an *ipv6 address* and a weight value to specify a weight that the device adds to routes that are received from the specified BGP4+ neighbor.

```
device(config-bgp-ipv6u)# neighbor 2001:db8:e0ff:783a::4 weight 200
```

The following example specifies a weight of 200 that the device adds to routes that are received from the specified BGP4+ neighbor.

```
device# configure terminal
device(config)# rbridge-id 122
device(config-rbridge-id-122)# router bgp
device(config-bgp-router)# local-as 65520
device(config-bgp-router)# address-family ipv6 unicast
device(config-bgp-ipv6u)# neighbor 2001:db8:e0ff:783a::4 weight 200
```

Enabling BGP4+ in a non-default VRF

When BGP4+ is enabled in a non-default VRF instance, the device enters BGP address-family IPv6 unicast VRF configuration mode. Several commands can then be accessed that allow the configuration of BGP4+ for the non-default VRF instance.

A non-default VRF instance has been configured.

1. Enter the **configure terminal** command to access global configuration mode.

   ```
   device# configure terminal
   ```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

   ```
   device(config)# rbridge-id 122
   ```

3. Enter the **router bgp** command to enable BGP routing globally.

   ```
   device(config-rbridge-id-122)# router bgp
   ```

4. Enter the **address-family ipv6 unicast** command with the **vrf** parameter, and specify a VRF name, to enter BGP address-family IPv6 unicast VRF configuration mode.

   ```
   device(config-bgp-router)# address-family ipv6 unicast vrf green
   ```

The following example enables BGP address-family IPv6 unicast VRF configuration mode where several commands can be accessed that allow the configuration of BGP4+ for the non-default VRF instance..

```
device# configure terminal
device(config)# rbridge-id 122
device(config-rbridge-id-122)# router bgp
device(config-bgp-router)# address-family ipv6 unicast vrf green
device(config-bgp-ipv6u-vrf)#
```

BGP4+ outbound route filtering

The BGP4+ Outbound Route Filtering Capability (ORF) feature is used to minimize the number of BGP updates sent between BGP peers.

When the ORF feature is enabled, unwanted routing updates are filtered out, reducing the amount of system resources required for generating and processing routing updates. The ORF feature is enabled through the advertisement of ORF capabilities to peer routers. The locally configured BGP4+ inbound prefix filters are sent to the remote peer so that the remote peer applies the filter as an outbound filter for the neighbor.

The ORF feature can be configured with send and receive ORF capabilities. The local peer advertises the ORF capability in send mode, indicating that it will accept a prefix list from a neighbor and apply the prefix list to locally configured ORFs. The local peer exchanges the ORF capability in send mode with a remote peer for a prefix list that is configured as an inbound filter for that peer locally. The remote

peer only sends the first update once it receives a ROUTEREFRESH request or BGP ORF with IMMEDIATE from the peer. The local and remote peers exchange updates to maintain the ORF on each router.

Configuring BGP4+ outbound route filtering

The BGP4+ Outbound Route Filtering (ORF) prefix list capability can be configured in receive mode, send mode, or both send and receive modes, minimizing the number of BGP updates exchanged between BGP peers.

1. Enter the **configure terminal** command to access global configuration mode.

```
device# configure terminal
```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

```
device(config)# rbridge-id 122
```

3. Enter the **router bgp** command to enable BGP routing.

```
device(config-rbridge-id-122)# router bgp
```

4. Enter the **address-family ipv6 unicast** command to enter IPv6 address family configuration mode.

```
device(config-bgp-router)# address-family ipv6 unicast
```

5. Enter the **neighbor activate** command, specifying an IPv6 address, to add a neighbor.

```
device(config-bgp-ipv6u)# neighbor 2001:db8:e0ff:783a::4 activate
```

6. Enter the **neighbor prefix-list** command, specify an IPv6 address and the **in** keyword to filter the incoming route updates from a specified BGP neighbor.

```
device(config-bgp-ipv6u)# neighbor 2001:db8:e0ff:783a::4 prefix-list myprefixlist in
```

7. Do one of the following:
   - Enter the **neighbor capability orf prefixlist** command and specify the **send** keyword to advertise ORF send capabilities.

```
device(config-bgp-ipv6u)# neighbor 2001:db8:e0ff:783a::4 capability orf prefixlist send
```

   - Enter the **neighbor capability orf prefixlist** command and specify the **receive** keyword to advertise ORF receive capabilities.

```
device(config-bgp-ipv6u)# neighbor 2001:db8:e0ff:783a::4 capability orf prefixlist receive
```

   - Enter the **neighbor capability orf prefixlist** command to configure ORF capability in both send and receive modes.

```
device(config-bgp-ipv6u)# neighbor 2001:db8:e0ff:783a::4 capability orf prefixlist
```

The following example configures ORF in receive mode.

```
device# configure terminal
device(config)# rbridge-id 122
device(config-rbridge-id-122)# router bgp
device(config-bgp-router)# address-family ipv6 unicast
device(config-bgp-ipv6u)# neighbor 2001:db8:e0ff:783a::4 activate
device(config-bgp-ipv6u)# neighbor 2001:db8:e0ff:783a::4 capability orf prefixlist receive
```

The following example configures ORF in send mode.

```
device# configure terminal
device(config)# rbridge-id 122
device(config-rbridge-id-122)# router bgp
device(config-bgp-router)# address-family ipv6 unicast
device(config-bgp-ipv6u)# neighbor 2001:db8:e0ff:783a::4 activate
device(config-bgp-ipv6u)# neighbor 2001:db8:e0ff:783a::4 prefix-list myprefixlist in
device(config-bgp-ipv6u)# neighbor 2001:db8:e0ff:783a::4 capability orf prefixlist send
```

The following example configures ORF in both send and receive modes.

```
device# configure terminal
device(config)# rbridge-id 122
device(config-rbridge-id-122)# router bgp
device(config-bgp-router)# address-family ipv6 unicast
device(config-bgp-ipv6u)# neighbor 2001:db8:e0ff:783a::4 activate
device(config-bgp-ipv6u)# neighbor 2001:db8:e0ff:783a::4 prefix-list myprefixlist in
device(config-bgp-ipv6u)# neighbor 2001:db8:e0ff:783a::4 capability orf prefixlist
```

BGP4+ confederations

A large autonomous system (AS) can be divided into multiple subautonomous systems and grouped into a single BGP4+ confederation.

Each subautonomous system must be uniquely identified within the confederation AS by a subautonomous system number. Within each subautonomous system, all the rules of internal BGP (IBGP) apply. For example, all BGP routers inside the subautonomous system must be fully meshed. Although EBGP is used between subautonomous systems, the subautonomous systems within the confederation exchange routing information like IBGP peers. Next hop, Multi Exit Discriminator (MED), and local preference information is preserved when crossing subautonomous system boundaries. To the outside world, a confederation looks like a single AS.

The AS path list is a loop-avoidance mechanism used to detect routing updates leaving one subautonomous system and attempting to re-enter the same subautonomous system. A routing update attempting to re-enter a subautonomous system it originated from is detected because the subautonomous system sees its own subautonomous system number listed in the update's AS path.

Configuring BGP4+ confederations

BGP4+ confederations, composed of multiple subautonomous systems, can be created.

1. Enter the **configure terminal** command to access global configuration mode.

   ```
   device# configure terminal
   ```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

   ```
   device(config)# rbridge-id 122
   ```

3. Enter the **router bgp** command to enable BGP routing.

   ```
   device(config-rbridge-id-122)# router bgp
   ```

4. Enter the **local-as** command to configure the autonomous system number (ASN) in which your device resides.

   ```
   device(config-bgp-router)# local-as 65520
   ```

5. Enter the **confederation identifier** command and specify an ASN to configure a BGP confederation identifier.

   ```
   device(config-bgp-router)# confederation identifier 100
   ```

6. Enter the **confederation peers** command and specify as many ASNs as needed to list all BGP peers that will belong to the confederation.

```
device(config-bgp-router)# confederation peers 65520 65521 65522
```

The following example creates a confederation with the confederation ID “100” and adds three subautonomous systems to the confederation.

```
device# configure terminal
device(config)# rbridge-id 122
device(config-rbridge-id-122)# router bgp
device(config-bgp-router)# local-as 65520
device(config-bgp-router)# confederation identifier 100
device(config-bgp-router)# confederation peers 65520 65521 65522
```

BGP4+ extended community

The BGP4+ extended community feature filters routes based on a regular expression specified when a route has multiple community values in it.

A BGP community is a group of destinations that share a common property. Community information identifying community members is included as a path attribute in BGP UPDATE messages. You can perform actions on a group using community and extended community attributes to trigger routing decisions. All communities of a particular type can be filtered out, or certain values can be specified for a particular type of community. You can also specify whether a particular community is transitive or non-transitive across an autonomous system (AS) boundary.

An extended community is an 8-octet value and provides a larger range for grouping or categorizing communities. BGP extended community attributes are specified in RFC 4360.

You define the extended community list using the **ip extcommunity-list** command. The extended community can then be matched or applied to the neighbor through the route map. The route map must be applied on the neighbor to which routes need to carry the extended community attributes. The "send-community" should be enabled for the neighbor configuration to start including the attributes while sending updates to the neighbor.

Defining BGP4+ extended communities

In order to apply a BGP4+ extended community filter, a BGP4+ extended community filter must be defined.

1. Enter the **configure terminal** command to access global configuration mode.

```
device# configure terminal
```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

```
device(config)# rbridge-id 122
```

3. Enter the **ip extcommunity-list** command and specify a number to set a BGP extended community filter.

```
device(config-rbridge-id-122)# ip extcommunity-list 1 permit soo 123:2
```

4. Enter the **route-map** *name* command to create and define a route map and enter route-map configuration mode.

```
device(config-rbridge-id-122)# route-map extComRmap permit 10
```

Permits a matching pattern.

5. Enter the **match extcommunity** command and specify an extended community list.

```
device(config-route-map-extComRmap/permit/10)# match extcommunity 1
```

6. Enter the **exit** command.

```
device(config-route-map-extComRmap/permit/10)# exit
```

7. Enter the **route-map***name* command to define a route map and enter route-map configuration mode.

```
device(config-rbridge-id-122)# route-map sendExtComRmap permit 10
```

   Permits a matching pattern.

8. Enter the **set extcommunity** command and specify the **rt** *extcommunity value* keyword to specify the route target (RT) extended community attribute.

```
device(config-route-map-sendExtComRmap/permit/10)# set extcommunity rt 3:3
```

9. Enter the **set extcommunity** command and specify the **soo** *extcommunity value* keyword to the site of origin (SOO) extended community attribute.

```
device(config-route-map-sendExtComRmap/permit/10)# set extcommunity soo 2:2
```

The following example configures an extended community ACL called “extended”, defines a route map, and permits and sets a matching pattern.

```
device# configure terminal
device(config)# rbridge-id 122
device(config-rbridge-id-122)# ip extcommunity-list 1 permit soo 123:2
device(config-rbridge-id-122)# route-map extComRmap permit 10
device(config-route-map-extComRmap/permit/10)# match extcommunity 1
device(config-route-map-extComRmap/permit/10)# exit
device(config-rbridge-id-122)# route-map sendExtComRmap permit 10
device(config-route-map-sendExtComRmap/permit/10)# set extcommunity rt 3:3
device(config-route-map-sendExtComRmap/permit/10)# set extcommunity soo 2:2
```

Applying a BGP4+ extended community filter

A BGP4+ extended community filter can be applied .

BGP4+ communities must already be defined.

1. Enter the **configure terminal** command to access global configuration mode.

```
device# configure terminal
```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

```
device(config)# rbridge-id 122
```

3. Enter the **ip extcommunity-list** command and specify a number to set a BGP extended community filter.

```
device(config-rbridge-id-122)# ip extcommunity-list 1 permit rt 123:2
```

4. Enter the **router bgp** command to enable BGP routing.

```
device(config-rbridge-id-122)# router bgp
```

5. Enter the **local-as** command to configure the autonomous system number (ASN) in which your device resides.

```
device(config-bgp-router)# local-as 1000
```

6. Enter the **neighbor remote-as** command, specifying an IPv6 address, to specify the ASN in which the remote neighbor resides.

```
device(config-bgp-router)# neighbor fe80:4398:ab30:45de::1 remote-as 1001
```

7. Enter the **address-family ipv6 unicast** command to enter IPv6 address family configuration mode.

```
device(config-bgp-router)# address-family ipv6 unicast
```

8. Enter the **neighbor activate** command to enable the exchange of information with the neighbor.

```
device(config-bgp-ipv6u)# neighbor fe80:4398:ab30:45de::1 activate
```

9. Enter the **neighbor route-map** command and specify the **in** keyword to apply a route map to incoming routes.

```
device(config-bgp-ipv6u)# neighbor fe80:4398:ab30:45de::1 route-map in extComRmapt
```

10. Enter the **neighbor route-map** command and specify the **out** keyword to apply a route map to outgoing routes.

```
device(config-bgp-ipv6u)# neighbor fe80:4398:ab30:45de::1 route-map out sendExtComRmap
```

11. Enter the **neighbor send-community** command and specify the **both** keyword to enable the sending of standard and extended attributes in updates to the specified BGP neighbor.

```
device(config-bgp-ipv6u)# neighbor fe80:4398:ab30:45de::1 send-community both
```

The following example applies a BGP4+ extended community filter.

```
device# configure terminal
device(config)# rbridge-id 122
device(config-rbridge-id-122)# ip extcommunity-list 1 permit rt 123:2
device(config-rbridge-id-122)# router bgp
device(config-bgp-router)# local-as 1000
device(config-bgp-router)# neighbor fe80:4398:ab30:45de::1 remote-as 1001
device(config-bgp-router)# address-family ipv6 unicast
device(config-bgp-ipv6u)# neighbor fe80:4398:ab30:45de::1 activate
device(config-bgp-ipv6u)# neighbor fe80:4398:ab30:45de::1 route-map in extComRmap
device(config-bgp-ipv6u)# neighbor fe80:4398:ab30:45de::1 route-map out sendExtComRmap
device(config-bgp-ipv6u)# neighbor fe80:4398:ab30:45de::1 send-community both
```

BGP4+ graceful restart

BGP4+ graceful restart (GR) allows for restarts where BGP neighboring devices participate in the restart, helping to ensure that no route and topology changes occur in the network for the duration of the restart.

The GR feature provides a routing device with the capability to inform its neighbors when it is performing a restart.

When a BGP session is established, GR capability for BGP is negotiated by neighbors through the BGP OPEN message. If the neighbor also advertises support for GR, GR is activated for that neighbor session. If both peers do not exchange the GR capability, the session is not GR-capable. If the BGP session is lost, the BGP peer router, known as a GR helper, marks all routes associated with the device as “stale” but continues to forward packets to these routes for a set period of time. The restarting device also continues to forward packets for the duration of the graceful restart. When the graceful restart is complete, routes are obtained from the helper so that the device is able to quickly resume full operation.

When the GR feature is configured on a device, both helper router and restarting router functionalities are supported. It is not possible to disable helper functionality explicitly.

GR is disabled by default and can be enabled in both IPv4 and IPv6 address families. When the GR timer expires, the BGP RASlog message is triggered.

Configuring BGP4+ graceful restart

The graceful restart (GR) feature can be configured on a routing device, providing it with the capability to inform its neighbors when it is performing a restart.

NOTE
High availability (HA) requires GR to be enabled.

1. Enter the **configure terminal** command to access global configuration mode.

```
device# configure terminal
```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

```
device(config)# rbridge-id 122
```

3. Enter the **router bgp** command to enable BGP routing.

```
device(config-rbridge-id-122)# router bgp
```

4. Enter the **local-as** command to configure the autonomous system number (ASN) in which your device resides.

```
device(config-bgp-router)# local-as 1000
```

5. Enter the **neighbor remote-as** command, specifying an IPv6 address, to specify the autonomous system ASN in which the remote neighbor resides.

```
device(config-bgp-router)# neighbor 1000::1 remote-as 2
```

6. Enter the **address-family ipv6 unicast** command to enter IPv6 address family configuration mode.

```
device(config-bgp-router)# address-family ipv6 unicast
```

7. Enter the **neighbor activate** command to add a neighbor.

```
device(config-bgp-ipv6u)# neighbor 1000::1 activate
```

8. Enter the **graceful-restart** command to enable the graceful restart feature.

```
device(config-bgp-ipv6u)# graceful-restart
```

9. Do any of the following:
   - Enter the **graceful-restart** command using the **purge-time** keyword to overwrite the default purge-time value.

     ```
     device(config-bgp-ipv6u)# graceful-restart purge-time 300
     ```

   - Enter the **graceful-restart** command using the **restart-time** keyword to overwrite the default restart-time advertised to graceful restart-capable neighbors.

     ```
     device(config-bgp-ipv6u)# graceful-restart restart-time 180
     ```

   - Enter the **graceful-restart** command using the **stale-routes-time** keyword to overwrite the default amount of time that a helper device will wait for an EOR message from a peer.

     ```
     device(config-bgp-ipv6u)# graceful-restart stale-routes-time 100
     ```

The following example enables the graceful restart feature.

```
device# configure terminal
device(config)# rbridge-id 122
device(config-rbridge-id-122)# router bgp
device(config-bgp-router)# local-as 1
device(config-bgp-router)# neighbor 1000::1 remote-as 2
device(config-bgp-router)# address-family ipv6 unicast
device(config-bgp-ipv6u)# neighbor 1000::1 activate
device(config-bgp-ipv6u)# graceful-restart
```

The following example enables the graceful restart feature and sets the purge time to 300 seconds, overwriting the default value.

```
device# configure terminal
device(config)# rbridge-id 122
device(config-rbridge-id-122)# router bgp
device(config-bgp-router)# local-as 1
device(config-bgp-router)# neighbor 1000::1 remote-as 2
device(config-bgp-router)# address-family ipv6 unicast
device(config-bgp-ipv6u)# neighbor 1000::1 activate
device(config-bgp-ipv6u)# graceful-restart purge-time 300
```

The following example enables the graceful restart feature and sets the restart time to 180 seconds, overwriting the default value.

```
device# configure terminal
device(config)# rbridge-id 122
device(config-rbridge-id-122)# router bgp
device(config-bgp-router)# local-as 1
device(config-bgp-router)# neighbor 1000::1 remote-as 2
device(config-bgp-router)# address-family ipv6 unicast
device(config-bgp-ipv6u)# neighbor 1000::1 activate
device(config-bgp-ipv6u)# graceful-restart restart-time 180
```

The following example enables the graceful restart feature and sets the stale-routes time to 100 seconds, overwriting the default value.

```
device# configure terminal
device(config)# rbridge-id 122
device(config-rbridge-id-122)# router bgp
device(config-bgp-router)# local-as 1
device(config-bgp-router)# neighbor 1000::1 remote-as 2
device(config-bgp-router)# address-family ipv6 unicast
device(config-bgp-ipv6u)# neighbor 1000::1 activate
device(config-bgp-ipv6u)# graceful-restart stale-routes-time 100
```

Use the **clear ipv6 bgp neighbor** command with the **all** parameter for the changes to the graceful restart parameters to take effect immediately.

BGP add path overview

The BGP add path feature provides the ability for multiple paths for the same prefix to be advertised without the new paths implicitly replacing the previous paths. Path diversity is achieved rather than path hiding.

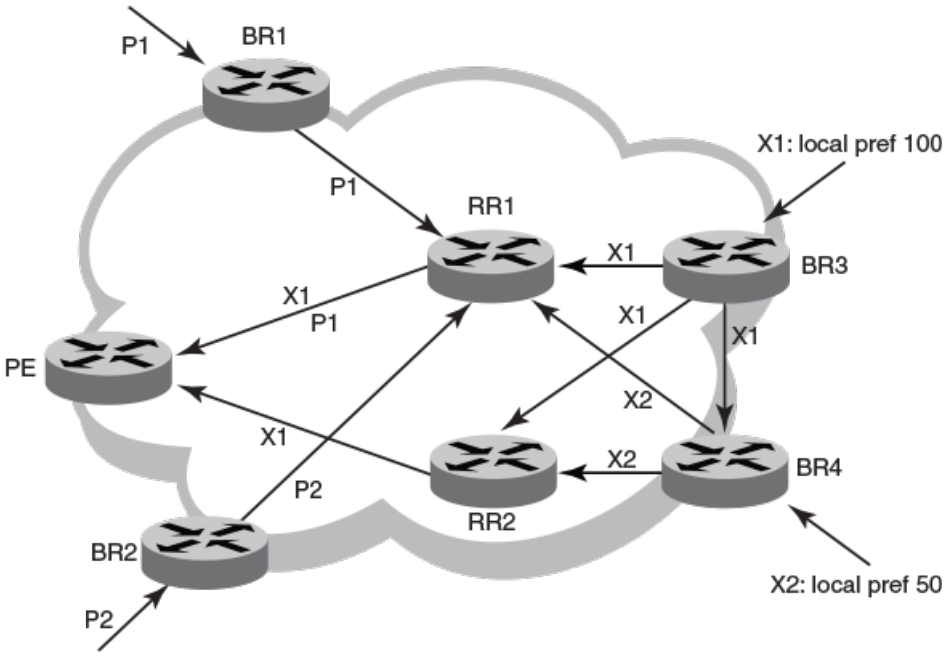
BGP devices and route reflectors (RRs) generally advertise only their best path to neighboring devices, even when multiple paths exist. The advertisement of the same prefix from the same neighbor replaces the previous announcement of that prefix. This is known as an implicit withdraw, behavior that achieves better scaling but at the cost of path diversity. When RRs are deployed inside an autonomous system (AS) to avoid iBGP scaling issues, only the best route is reflected even if multiple Equal-Cost Multipath (ECMP) routes exist. These secondary best paths remain hidden.

Path hiding can affect the efficient use of BGP multipath and path diversity, and prevent hitless planned maintenance. Upon next hop failures, path hiding also inhibits fast and local recovery because the network has to wait for BGP control plane convergence to restore traffic. BGP add path enables BGP to advertise even the secondary best routes so that multiple paths for the same prefix can be advertised without the new paths implicitly replacing previous paths. BGP add path provides a generic way of offering path diversity.

In the figure below, path hiding occurs in two ways:

- Prefix P has paths P1 and P2 advertised from BR1 and BR2 to RR1. RR1 selects P1 as the best path and advertises only P1 to PE.
- Prefix X has path X1 advertised from BR3 to BR4 with local preference 100. BR4 also has path X2. However, only the best path, X1, is selected. BR3 advertises X1 to the RRs and X2 is suppressed.

FIGURE 25 BGP path hiding



NOTE
BGP add path is supported for both route reflectors and confederations.

NOTE
BGP add path is supported for both VCS fabrics and IP fabrics, but is supported for the IPv4 and IPv6 unicast address families only.

Advantages and limitations of BGP add path

The following are the advantages and limitations provided by BGP add path.

Advantages of BGP add path

- **Fast convergence and fault tolerance**: When BGP add path is enabled, more than one path to a destination is advertised. If one of the paths goes down, connectivity is easily restored due to the availability of backup paths. If the next hop for the prefix becomes unreachable, the device can switch to the backup route immediately without having to wait for BGP control plane messages.
- **Enhanced load balancing capabilities**: Traditionally with RRs in an iBGP domain, only the best path is given to the clients even if ECMP paths exists. This affects load balancing. With additional paths advertised by RRs, the clients have more effective load balance.

Limitations of BGP add path

- Load balancing is achieved for the BGP IPv4 and IPv6 address family neighbors only and not for EVPN neighbors. For more information on BGP EVPN, refer to the BGP EVPN chapter.
- BGP add path does not provide any advantage for ARP and MAC routes supported by EVPN peers. L2 ECMP is not currently supported.
- BGP multipath must be configured to choose ECMP paths. Only the best path and the first 5 ECMP paths are chosen as additional paths for basic functionality support and advertised to neighbors with path IDs.

BGP add path functionality

BGP add path is implemented by including an additional four octet value known as a path identifier (ID) for each path in the NLRI. Path IDs are unique to a peering session and are generated for each network.

A path ID can apply to the IPv4 or IPv6 unicast address family.

Therefore, when the same prefix is received with the same path ID from the same peer, it is considered as a replacement route or a duplicate route. When the same prefix is received with a different path ID from the same peer, it is considered as an additional path to the prefix.

To send or receive additional paths, the add path capability must be negotiated. If it isn't negotiated, only the best path can be sent. BGP updates carry the path ID once the add path capability is negotiated. In order to carry the path ID in an update message, the existing NLRI encodings are extended by prepending the path ID field, which consists of four octets.

The assignment of the path ID for a path by a BGP device occurs in such a way that the BGP device is able to use the prefix and path ID to uniquely identify a path advertised to a neighbor so as to continue to send further updates for that path. The receiving BGP neighbor that re-advertises a route regenerates its own path ID to be associated with the re-advertised route.

The set of additional paths advertised to each neighbor can be different, and advertise filters are provided on a per neighbor basis.

Negotiating BGP4+ add paths capability

BGP4+ neighbors can be configured to send or receive additional paths after negotiation is completed.

1. Enter the **configure terminal** command to access global configuration mode.

```
device# configure terminal
```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

```
device(config)# rbridge-id 122
```

3. Enter the **router bgp** command to enable BGP routing.

```
device(config-rbridge-id-122)# router bgp
```

4. Enter the **local-as** command to configure the autonomous system number (ASN) in which your device resides.

```
device(config-bgp-router)# local-as 65520
```

5. Enter the **neighbor** *ipv6 address* **remote-as** command to add a neighbor.

```
device(config-bgp-router)# neighbor 2001:2018:8192::125 remote-as 2
```

6. Enter the **address-family unicast** command using the **ipv6** parameter to enter BGP address-family IPv6 unicast configuration mode.

```
device(config-bgp-router)# address-family ipv6 unicast
```

7. Enter the **neighbor capability additional-paths** command, specifying an IPv6 address, to enable the advertisement of additional paths. Enter the **receive** parameter to enable BGP4+ to receive additional paths from BGP neighbors.

```
device(config-bgp-ipv6u)# neighbor 2001:2018:8192::125 capability additional-paths receive
```

8. Enter the **neighbor capability additional-paths** command, specifying an IPv6 address, to enable the advertisement of additional paths. Enter the **send** parameter to enable BGP4+ to send additional paths to BGP neighbors.

```
device(config-bgp-ipv6u)# neighbor 2001:2018:8192::125 capability additional-paths send
```

The following example enables BGP4+ to receive additional paths from BGP neighbors and to send additional paths to BGP neighbors.

```
device# configure terminal
device(config)# rbridge-id 122
device(config-rbridge-id-122)# router bgp
device(config-bgp-router)# local-as 65520
device(config-bgp-router)# neighbor 2001:2018:8192::125 remote-as 2
device(config-bgp-router)# address-family ipv6 unicast
device(config-bgp-ipv6u)# neighbor 2001:2018:8192::125 capability additional-paths receive
device((config-bgp-ipv6u))# neighbor 2001:2018:8192::125 capability additional-paths send
```

Advertising best BGP4+ additional paths

Additional paths that the device selects as best paths can be advertised.

1. Enter the **configure terminal** command to access global configuration mode.

```
device# configure terminal
```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

```
device(config)# rbridge-id 122
```

3. Enter the **router bgp** command to enable BGP routing.

```
device(config-rbridge-id-122)# router bgp
```

4. Enter the **local-as** command to configure the autonomous system number (ASN) in which your device resides.

```
device(config-bgp-router)# local-as 65520
```

5. Enter the **neighbor** *ipv6 address* **remote-as** command to add a neighbor.

```
device(config-bgp-router)# neighbor 2001:2018:8192::125 remote-as 2
```

6. Enter the **address-family unicast** command using the **ipv6** parameter to enter BGP address-family IPv6 unicast configuration mode.

```
device(config-bgp-router)# address-family ipv6 unicast
```

7. Enter the **neighbor additional-paths advertise** command, specifying an IPv6 address, using the **best** *value* parameter to configure BGP4+ to advertise the best BGP path and an additional two routes to this neighbor.

```
device(config-bgp-ipv6u)# neighbor 2001:2018:8192::125 additional-paths advertise best 2
```

The following example configures BGP4+ to advertise the best BGP path and an additional two routes a to a BGP neighbor.

```
device# configure terminal
device(config)# rbridge-id 122
device(config-rbridge-id-122)# router bgp
device(config-bgp-router)# local-as 65520
device(config-bgp-router)# neighbor 2001:2018:8192::125 remote-as 2
device(config-bgp-router)# address-family ipv6 unicast
device(config-bgp-ipv6u)# neighbor 2001:2018:8192::125 additional-paths advertise best 2
```

Advertising all BGP4+ additional paths

Additional BGP paths can be advertised.

1. Enter the **configure terminal** command to access global configuration mode.

```
device# configure terminal
```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

```
device(config)# rbridge-id 122
```

3. Enter the **router bgp** command to enable BGP routing.

```
device(config-rbridge-id-122)# router bgp
```

4. Enter the **local-as** command to configure the autonomous system number (ASN) in which your device resides.

```
device(config-bgp-router)# local-as 65520
```

5. Enter the **neighbor** *ipv6 address* **remote-as** command to add a neighbor.

```
device(config-bgp-router)# neighbor 2001:2018:8192::125 remote-as 2
```

6. Enter the **address-family unicast** command using the **ipv6** parameter to enter BGP address-family IPv6 unicast configuration mode.

```
device(config-bgp-router)# address-family ipv6 unicast
```

7. Enter the **neighbor additional-paths advertise** command, specifying an IPv6 address, using the **all** parameter to configure BGP4+ to advertise all BGP4+ paths with a unique next hop.

```
device(config-bgp-ipv6u)# neighbor 2001:2018:8192::125 additional-paths advertise all
```

The following example configures BGP4+ to advertise all BGP additional paths.

```
device# configure terminal
device(config)# rbridge-id 122
device(config-rbridge-id-122)# router bgp
device(config-bgp-router)# local-as 65520
device(config-bgp-router)# neighbor 2001:2018:8192::125 remote-as 2
device(config-bgp-router)# address-family ipv6 unicast
device(config-bgp-ipv6u)# neighbor 2001:2018:8192::125 additional-paths advertise all
```

Configuring path based filtering using outbound filters

In the below task when the additional paths are advertised to neighbor 10.0.0.1, only the paths that have next hops in the network 10.1.0.0/24 and 10.2.0.0/24 are permitted, and all other paths for the prefixes are denied.

1. Enter the **configure terminal** command to access global configuration mode.

```
device# configure terminal
```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

```
device(config)# rbridge-id 122
```

3. Enter the **ip prefix-list** command in RBridge ID configuration mode and enter a name to configure an IP prefix list.

```
device(config-rbridge-id-122)# ip prefix-list sample seq 1 permit 10.1.0.0/24
```

   The prefix list name and sequence number are specified and the prefix 10.1.0.0/24 is permitted.

4. Enter the **ip prefix-list** command in RBridge ID configuration mode and enter a different name to configure an IP prefix list.

```
device(config-rbridge-id-122)# ip prefix-list sample seq 2 permit 10.2.0.0/24
```

   The prefix list name and sequence number are specified and the prefix 10.2.0.0/24 is permitted.

5. Enter the **route-map** *name* **permit** command to define the route map and enter route map configuration mode.

```
device(config-rbridge-id-122)# route-map test permit 1
```

6. Enter the **match ip address** command and specify the name of a prefix list.

```
device(config-route-map-test/permit/1)# match ip next-hop prefix-list sample
```

7. Enter the **exit** command to return to RBridge ID configuration mode.

```
device(config-route-map-test/permit/1)# exit
```

8. Enter the **router bgp** command to enable BGP routing.

```
device(config-rbridge-id-122)# router bgp
```

9. Enter the **address-family unicast** command using the **ipv4** parameter to enter BGP address-family IPv4 unicast configuration mode.

```
device(config-bgp-router)# address-family ipv4 unicast
```

10. Enter the **neighbor capability additional-paths** command, specifying an IP address, to enable the advertisement of additional paths.

```
device(config-bgp-ipv4u)# neighbor 10.0.0.1 capability additional-paths
```

11. Enter the **neighbor additional-paths advertise** command, specifying an IP address, using the **best** *value* parameter to configure BGP4 to advertise best BGP4 additional paths.

```
device(config-bgp-ipv4u)# neighbor 10.0.0.1 additional-paths advertise best 3
```

12. Enter the **neighbor** *ip-address* **route-map** command using the **out** keyword to apply a route map to outgoing routes.

```
device(config-bgp-ipv6u)# neighbor 10.0.0.1 route-map out test
```

The following example configures BGP4 to advertise the 3 best BGP additional paths to neighbor 10.0.0.1. Only the paths that have next hops in the network 10.1.0.0/24 and 10.2.0.0/24 are permitted and all other paths for the prefixes are denied.

```
device# configure terminal
device(config)# rbridge-id 122
device(config-rbridge-id-122)# ip prefix-list sample seq 1 permit 10.1.0.0/24
device(config-rbridge-id-122)# ip prefix-list sample seq 2 permit 10.2.0.0/24
device(config-rbridge-id-122)# route-map test permit 1
device(config-route-map-test/permit/1)# match ip next-hop prefix-list sample
device(config-route-map-test/permit/1)# exit
device(config-rbridge-id-122)# router bgp
device(config-bgp-router)# address-family ipv4 unicast
device(config-bgp-ipv4u)# neighbor 10.0.0.1 capability additional-paths
device(config-bgp-ipv4u)# neighbor 10.0.0.1 additional-paths advertise best 3
device(config-bgp-ipv6u)# neighbor 10.0.0.1 route-map out test
```

Auto shutdown of BGP neighbors on initial configuration

The auto shutdown of BGP neighbors on initial configuration feature prevents a BGP peer from attempting to establish connections with remote peers when the BGP peer is first configured. Sessions are only initiated when the entire configuration for the BGP peer is complete.

When the auto shutdown of BGP neighbors on initial configuration feature is enabled, the value of the shutdown parameter for any existing configured neighbor is not changed. Any new BGP neighbor configured after the feature is enabled has the shutdown state set to the configured value. When the feature is enabled and a new BGP neighbor is configured, the shutdown parameter of the BGP neighbor is automatically set to enabled. When all the BGP neighbor parameters are configured and it is ready to start the establishment of BGP session with the remote peer, the shutdown parameter of the BGP neighbor is then disabled.

Any new neighbor configured and added to a peer group has the shutdown flag enabled by default. Additionally, any new neighbor configured with the autonomous system (AS) in which that remote neighbor resides specified using the **neighbor remote-as** command has the shutdown flag enabled by default.

Configuring auto shutdown of BGP neighbors on initial configuration

Follow this procedure to enable auto shutdown of BGP neighbors on initial configuration.

1. Enter the **configure terminal** command to access global configuration mode.

```
device# configure terminal
```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

```
device(config)# rbridge-id 122
```

3. Enter the **router bgp** command to enable BGP routing.

```
device(config-rbridge-id-122)# router bgp
```

4. Enter the **local-as** command to configure the autonomous system number (ASN) in which your device resides.

```
device(config-bgp-router)# local-as 65520
```

5. Enter the **auto-shutdown-new-neighbors** command to prevent the BGP peer from attempting to establish connections with remote peers until all BGP neighbor parameters are configured.

```
device(config-bgp-router)# auto-shutdown-new-neighbors
```

The following example enables auto shutdown of BGP neighbors on initial configuration.

```
device# configure terminal
device(config)# rbridge-id 122
device(config-rbridge-id-122)# router bgp
device(config-bgp-router)# local-as 65520
device(config-bgp-router)# auto-shutdown-new-neighbors
```

Disabling the BGP4+ peer shutdown state

When the auto shutdown of BGP4+ neighbors on initial configuration feature is enabled, a BGP4+ peer is prevented from attempting to establish connections with remote peers when the BGP4+ peer is first configured. Once all of the configuration parameters for the peer are complete, you can start the BGP4+ session establishment process and disable the peer shutdown state. Follow this procedure to disable the peer shutdown state.

1. Enter the **configure terminal** command to access global configuration mode.

```
device# configure terminal
```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

```
device(config)# rbridge-id 122
```

3. Enter the **router bgp** command to enable BGP routing.

```
device(config-rbridge-id-122)# router bgp
```

4. Enter the **local-as** command to configure the autonomous system number (ASN) in which your device resides.

```
device(config-bgp-router)# local-as 65520
```

5. Enter the **no neighbor shutdown** command, specifying an IPv6 address, to disable the peer shutdown state and begin the BGP4+ session establishment process.

```
device(config-bgp-router)# no neighbor 2001:2018:8192::125 shutdown
```

The following example disables the peer shutdown state and begins the BGP4+ session establishment process.

```
device# configure terminal
device(config)# rbridge-id 122
device(config-rbridge-id-122)# router bgp
device(config-bgp-router)# local-as 65520
device(config-bgp-router)# no neighbor 2001:2018:8192::125 shutdown
```

BGP4+ graceful shutdown

BGP4+ graceful shutdown automatically reroutes traffic to an alternate path prior to shutting down a neighbor, minimizing traffic loss during a planned shutdown and reestablishment of BGP4+ sessions.

When a planned maintenance is carried out, traffic loss is generally expected until BGP converges in the network. BGP graceful shutdown provides benefits, such as fewer lost packets and less time spent reconfiguring devices by reducing the loss of inbound or outbound traffic flows forwarded along a peering link that is being shut down for planned maintenance.

BGP graceful shutdown helps to mitigate traffic loss so that traffic is automatically rerouted while the neighbor is shutdown.

When BGP graceful shutdown is initiated on one or more BGP neighbors, BGP starts a graceful shutdown timer on the graceful shutdown initiator and then performs a route refresh to all graceful shutdown neighbors with updated BGP attributes, such as LOCAL_PREFRENCE or confederations for iBGP, or AS_PATH prepend for eBGP. The best path calculation on all the graceful shutdown is then triggered, making the routes via the graceful shutdown initiator less preferable. Thus, the traffic from graceful shutdown neighbors is re-routed to alternate paths. However, the path via the graceful shutdown initiator is still valid until the graceful shutdown timer expires. Once the graceful shutdown timer expires, all the graceful shutdown neighbors are shut down on the graceful shutdown initiator and all traffic is re-routed to alternate paths. The operator can now carry the maintenance operation.

For the graceful shutdown initiator, if there is any outbound policy associated with a particular graceful shutdown neighbor, the outbound policy is applied before the graceful shutdown parameters are applied.

NOTE

For the graceful shutdown neighbor, if there is any inbound policy associated with the graceful shutdown initiator modifying the LOCAL_PREFERENCE or ASPATH, the graceful shutdown parameters become insignificant. In order to prevent this, it is recommended to modify the existing inbound policy rule to match the graceful shutdown community attribute and make the route less preferred. Therefore, the send-community attribute should also be negotiated between graceful shutdown initiator and neighbor.

Configuring graceful shutdown for all BGP4+ neighbors

The graceful shutdown feature can be configured on a routing device so that traffic is automatically rerouted to an alternate path prior to shutting down a BGP4+ neighbor, minimizing traffic loss during a planned shutdown and reestablishment of BGP4+ sessions. A specific BGP neighbor or all BGP neighbors can be gracefully shut down. The following task gracefully shuts down all BGP4+ neighbors.

1. Enter the **configure terminal** command to access global configuration mode.

```
device# configure terminal
```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

```
device(config)# rbridge-id 122
```

3. Enter the **router bgp** command to enable BGP routing.

```
device(config-rbridge-id-122)# router bgp
```

4. Enter the **local-as** command to configure the autonomous system number (ASN) in which your device resides.

```
device(config-bgp-router)# local-as 1000
```

5. Enter the **neighbor** *ipv6-address* **remote-as** command to add a neighbor.

```
device(config-bgp-router)# neighbor 1000::1 remote-as 2
```

6. Enter the **graceful-shutdown** command with the **community** parameter to gracefully shut down all BGP neighbors, set the graceful shutdown time, and set the community attribute.

```
device(config-bgp-router)# graceful-shutdown 600 community 10
```

The following example gracefully shuts down all BGP neighbors and sets the graceful shutdown timer to 600 seconds. The community attribute is set to 10.

```
device# configure terminal
device(config)# rbridge-id 122
device(config-rbridge-id-122)# router bgp
device(config-bgp-router)# local-as 1
device(config-bgp-router)# neighbor 1000::1 remote-as 2
device(config-bgp-router)# graceful-shutdown 600 community 10
```

Configuring graceful shutdown for a BGP4+ peer group

The graceful shutdown feature can be configured on a routing device so that traffic is automatically rerouted to an alternate path prior to shutting down a BGP4+ peer group, minimizing traffic loss during a planned shutdown and reestablishment of BGP4+ sessions. The following task gracefully shuts down a BGP4+ peer group.

1. Enter the **configure terminal** command to access global configuration mode.

```
device# configure terminal
```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

```
device(config)# rbridge-id 122
```

3. Enter the **router bgp** command to enable BGP routing.

```
device(config-rbridge-id-122)# router bgp
```

4. Enter the **local-as** command to configure the autonomous system number (ASN) in which your device resides.

```
device(config-bgp-router)# local-as 1000
```

5. Enter the **neighbor** *peer-group-name* **peer-group** command to add a peer group.

```
device(config-bgp-router)# neighbor grp-1 peer-group
```

6. Enter the **neighbor** *peer-group* **graceful-shutdown** command and specify a value to gracefully shut down the BGP4 peer group and set the graceful shutdown timer. Use the **community** parameter to set the community attribute.

```
device(config-bgp-router)# neighbor grp-1 graceful-shutdown 600 community 20
```

The following example gracefully shuts down a BGP4+ peer group and sets the graceful shutdown timer to 600 seconds. The community attribute is set to 3000 and the local preference attribute is set to 20.

```
device# configure terminal
device(config)# rbridge-id 122
device(config-rbridge-id-122)# router bgp
device(config-bgp-router)# local-as 1
device(config-bgp-router)# neighbor grp-1 peer-group
device(config-bgp-router)# neighbor grp-1 graceful-shutdown 600 community 20
```

Configuring graceful shutdown for a specified BGP4+ neighbor

The graceful shutdown feature can be configured on a routing device so that traffic is automatically rerouted to an alternate path prior to shutting down a BGP4+ neighbor, minimizing traffic loss during a planned shutdown and reestablishment of BGP4+ sessions. A specific BGP4+ neighbor or all BGP4+ neighbors can be gracefully shut down. The following task gracefully shuts down a specified BGP4+ neighbor.

1. Enter the **configure terminal** command to access global configuration mode.

```
device# configure terminal
```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

```
device(config)# rbridge-id 122
```

3. Enter the **router bgp** command to enable BGP routing.

```
device(config-rbridge-id-122)# router bgp
```

4. Enter the **local-as** command to configure the autonomous system number (ASN) in which your device resides.

```
device(config-bgp-router)# local-as 1000
```

5. Enter the **neighbor** *ipv6-address* **remote-as** command to add a neighbor.

```
device(config-bgp-router)# neighbor 1000::1 remote-as 2
```

6. Enter the **neighbor** *ipv6-address* **graceful-shutdown** command and specify a value to gracefully shut down the BGP4 neighbor and set the graceful shutdown timer. Use the **community** parameter to set the community attribute.

```
device(config-bgp-router)# neighbor 1000::1 graceful-shutdown 600 commuity 30
```

The following example gracefully shuts down a specified BGP4+ neighbor and sets the graceful shutdown timer to 600 seconds. The community attribute is set to 30.

```
device# configure terminal
device(config)# rbridge-id 122
device(config-rbridge-id-122)# router bgp
device(config-bgp-router)# local-as 1
device(config-bgp-router)# neighbor 1000::1 remote-as 2
device(config-bgp-router)# neighbor 1000::1 graceful-shutdown 600 community 30
```

Generalized TTL Security Mechanism support

Generalized TTL Security Mechanism (GTSM) is a lightweight security mechanism that protects external Border Gateway Protocol (eBGP) peering sessions from CPU utilization-based attacks using forged IP packets. GTSM prevents attempts to hijack the eBGP peering session by a host on a network segment that is not part of either BGP network, or by a host on a network segment that is not between the eBGP peers.

GTSM is enabled by configuring a minimum Time To Live (TTL) value for incoming IP packets received from a specific eBGP peer. BGP establishes and maintains the session only if the TTL value in the IP packet header is equal to or greater than the TTL value configured for the peering session. If the value is less than the configured value, the packet is silently discarded and no Internet Control Message Protocol (ICMP) message is generated.

In the case of directly connected neighbors, the device expects the BGP control packets received from the neighbor to have a TTL value of either 254 or 255. For multihop peers, the device expects the TTL for BGP control packets received from the neighbor to be greater than or equal to 255, minus the configured number of hops to the neighbor. If the BGP control packets received from the neighbor do not have the expected value, the device drops them.

Assumptions and limitations

- GTSM is supported for both directly connected peering sessions and multihop eBGP peering sessions.
- GTSM is supported for eBGP only.
- GTSM does not protect the integrity of data sent between eBGP peers and does not validate eBGP peers through any authentication method.
- GTSM validates only the locally configured TTL count against the TTL field in the IP packet header.
- GTSM should be configured on each participating device to maximize the effectiveness of this feature.
- When GTSM is enabled, the eBGP session is secured in the incoming direction only and has no effect on outgoing IP packets or the remote device.

Configuring GTSM for BGP4+

Generalized TTL Security Mechanism (GTSM) can be configured to protect external Border Gateway Protocol (eBGP) peering sessions .

1. Enter the **configure terminal** command to access global configuration mode.

```
device# configure terminal
```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

```
device(config)# rbridge-id 122
```

3. Enter the **router bgp** command to enable BGP routing.

```
device(config-rbridge-id-122)# router bgp
```

4. Enter the **local-as** command to configure the autonomous system number (ASN) in which your device resides.

```
device(config-bgp-router)# local-as 65520
```

5. Enter the **neighbor remote-as** command, specifying an IPv6 address, to add a neighbor.

```
device(config-bgp-router)# neighbor 2001:2018:8192::125 remote-as 2
```

6. Enter the **neighbor ebgp-btsh** command, specifying an IPv6 address, to enable GTSM.

```
device(config-bgp-router)# neighbor 2001:2018:8192::125 ebgp-btsh
```

The following example enables GTSM between a device and a neighbor with the IPv6 address 2001:2018:8192::125.

```
device# configure terminal
device(config)# rbridge-id 122
device(config-rbridge-id-122)# router bgp
device(config-bgp-router)# local-as 65520
device(config-bgp-router)# neighbor 2001:2018:8192::125 remote-as 2
device(config-bgp-router)# neighbor 2001:2018:8192::125 ebgp-btsh
```

BGP VRF route filters

Filtering based on route-map policies can be added to the EVPN IPv4 and IPv6 prefix routes that are imported to or exported from VRFs.

Refer to the topic "BGP VRF route filters" and the task "Using BGP VRF route filters" in the chapter "BGP" in this document.

Disabling the BGP AS_PATH check function

A device can be configured so that the AS_PATH check function for routes learned from a specific location is disabled, and routes that contain the recipient BGP speaker's AS number are not rejected.

1. Enter the **configure terminal** command to access global configuration mode.

```
device# configure terminal
```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

```
device(config)# rbridge-id 122
```

3. Enter the **router bgp** command to enable BGP routing.

```
device(config-rbridge-id-122)# router bgp
```

4. Enter the **address-family ipv6 unicast** command to enter IPv6 address family unicast configuration mode.

```
device(config-bgp-router)# address-family ipv6 unicast
```

5. Enter the **neighbor allowas-in** command with an IPv6 address and specify a **number** to disable the BGP AS_PATH check function, and specify the number of times that the AS path of a received route may contain the recipient BGP speaker's AS number and still be accepted.

```
device(config-bgp-ipv6u)# neighbor 2001:db8:e0ff:783a::4 allowas-in 3
```

The following example specifies that the AS path of a received route may contain the recipient BGP speaker's AS number three times and still be accepted.

```
device# configure terminal
device(config)# rbridge-id 122
device(config-rbridge-id-122)# router bgp
device(config-bgp-router)# address-family ipv6 unicast
device(config-bgp-ipv6u)# neighbor 2001:db8:e0ff:783a::4 allowas-in 3
```

Displaying BGP4+ statistics

Various **show ipv6 bgp** commands verify information about BGP4+ configurations.

Use one or more of the following commands to verify BGP4+ information. The commands do not have to be entered in this order.

1. Enter the **show ipv6 bgp summary** command.

```
device# show ipv6 bgp summary

  BGP4 Summary
  Router ID: 122.122.122.122   Local AS Number: 122
  Confederation Identifier: not configured
  Confederation Peers:
  Cluster ID: 122
  Maximum Number of IP ECMP Paths Supported for Load Sharing: 1
  Number of Neighbors Configured: 20, UP: 15
  Number of Routes Installed: 219, Uses 20805 bytes
  Number of Routes Advertising to All Neighbors: 2802 (440 entries), Uses 26400 bytes
  Number of Attribute Entries Installed: 31, Uses 2852 bytes
  Neighbor Address  AS#         State     Time     Rt:Accepted Filtered Sent     ToSend
  2001:54:54::54    122         ESTAB     0h19m58s    0        0        146      0
  2001:55:55::55    122         ESTAB     0h19m54s    1        0        146      0
  2001:122:53::53   6000        ESTAB     0h22m39s    50       0        147      0
  2001:122:534:2::534
                    534         ESTAB     0h 3m20s    10       0        137      0
  2001:125:125::125 122         CONN      0h11m33s    0        0        0       -
```

   This example output gives summarized BGP4+ information.

2. Enter the **show ipv6 bgp attribute-entries** command.

```
device# show ipv6 bgp attribute-entries

        Total number of BGP Attribute Entries: 2
1       Next Hop   : 2001::1                                MED      :1              Origin:IGP
        Originator:0.0.0.0           Cluster List:None
        Aggregator:AS Number :0           Router-ID:0.0.0.0         Atomic:None
        Local Pref:1              Communities:Internet
        AS Path    : (length 0)
        Address: 0x1205c75c  Hash:268 (0x01000000)
        Links: 0x00000000, 0x00000000
        Reference Counts: 2:0:0, Magic: 1
2       Next Hop   : ::                                     MED      :1              Origin:IGP
        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    : (length 0)
           AsPathLen: 0  AsNum: 0, SegmentNum: 0, Neighboring As: 0, Source As 0
        Address: 0x1205c7cc  Hash:365 (0x01000000)
        Links: 0x00000000, 0x00000000
        Reference Counts: 1:0:1, Magic: 2
```

   This example shows information about two route-attribute entries that are stored in device memory.

3. Enter the **show ipv6 bgp peer-group** command.

```
device# show ipv6 bgp peer-group

1 BGP peer-group is P1, Remote AS: 1
Address family : IPV4 Unicast
activate
Address family : IPV4 Multicast
no activate
Address family : IPV6 Unicast
activate
Address family : IPV6 Multicast
no activate
Address family : VPNV4 Unicast
no activate
Address family : L2VPN VPLS
no activate
Members:
IP Address: 2001::1
IP Address: 2001:0:0:1::1
IP Address: 10.1.0.1
```

This example shows output for a peer group called “P1”.

4. Enter the **show ipv6 bgp routes** command.

```
device# show ipv6 bgp routes
Total number of BGP Routes: 6
Status A:AGGREGATE B:BEST b:NOT-INSTALLED-BEST C:CONFED_EBGP D:DAMPED
       E:EBGP H:HISTORY I:IBGP L:LOCAL M:MULTIPATH m:NOT-INSTALLED-MULTIPATH
       S:SUPPRESSED F:FILTERED s:STALE
       Prefix             Next Hop        MED        LocPrf     Weight Status
1      57:7000:3:22:abc:1::/128  2001:700:122:57::57
                                                     100        0      BE
         AS_PATH: 7000 322
2      57:7000:3:22:abc:1:0:2/128  2001:700:122:57::57
                                                     100        0      BE
         AS_PATH: 7000 322
3      57:7000:3:22:abc:1:0:4/128  2001:700:122:57::57
                                                     100        0      BE
         AS_PATH: 7000 322
4      57:7000:3:22:abc:1:0:6/128  2001:700:122:57::57
                                                     100        0      BE
         AS_PATH: 7000 322
5      57:7000:3:22:abc:1:0:8/128  2001:700:122:57::57
                                                     100        0      BE
         AS_PATH: 7000 322
6      57:7000:3:22:abc:1:0:a/128  2001:700:122:57::57
                                                     100        0      BE
         AS_PATH: 7000 322
```

This example shows general BGP4+ route information.

5. Enter the **show ipv6 bgp routes** command, using the **summary** keyword.

```
device# show ipv6 bgp routes summary

  Total number of BGP routes (NLRIs) Installed     : 558
  Distinct BGP destination networks                : 428
  Filtered bgp routes for soft reconfig            : 0
  Routes originated by this router                 : 19
  Routes selected as BEST routes                   : 417
  BEST routes not installed in IP forwarding table : 0
  Unreachable routes (no IGP route for NEXTHOP)    : 22
  IBGP routes selected as best routes              : 102
  EBGP routes selected as best routes              : 296
```

This example shows summarized BGP4+ route information.

6. Enter the **show ipv6 bgp routes** command, using the **local** keyword.

```
device# show ipv6 bgp routes local
Status A:AGGREGATE B:BEST b:NOT-INSTALLED-BEST C:CONFED_EBGP D:DAMPED
       E:EBGP H:HISTORY I:IBGP L:LOCAL M:MULTIPATH m:NOT-INSTALLED-MULTIPATH
       S:SUPPRESSED F:FILTERED s:STALE
       Prefix             Next Hop        MED        LocPrf     Weight Status
1      131::1/128         ::              1          100        32768  BL
         AS_PATH:
2      2001:107:6133:2007:1::/112  2001:2007::201
                                          107        100        32768  BL
         AS_PATH:
3      2001:107:6133:2007:2::/112  2001:2007::202
                                          107        100        32768  BL
         AS_PATH:
4      2001:107:6133:2007:3::/112  2001:2007::203
                                          107        100        32768  BL
         AS_PATH:
5      2001:107:6133:2007:4::/112  2001:2007::204
                                          107        100        32768  BL
         AS_PATH:
6      2001:107:6133:2007:5::/112  2001:2007::205
                                          107        100        32768  BL
         AS_PATH:
7      2001:107:6133:2007:6::/112  2001:2007::206
                                          107        100        32768  BL
```

This example shows information about local routes.

Displaying BGP4+ neighbor statistics

Various **show ipv6 bgp neighbor** commands verify information about BGP4+ neighbor configurations.

Use one or more of the following commands to verify BGP4+ neighbor information. The commands do not have to be entered in this order.

1. Enter the **show ipv6 bgp neighbors** command.

```
device# show ipv6 bgp neighbors
Total number of BGP Neighbors: 2
IP Address: 2001::1, AS: 2 (EBGP), RouterID: 192.0.0.1, VRF: default-vrf
State: ESTABLISHED, Time: 0h0m27s, KeepAliveTime: 30, HoldTime: 90
KeepAliveTimer Expire in 3 seconds, HoldTimer Expire in 62 seconds
Minimal Route Advertisement Interval: 0 seconds
Messages: Open Update KeepAlive Notification Refresh-Req
Sent : 5 2 7 3 0
Received: 5 4 11 1 0
Last Update Time: NLRI Withdraw NLRI Withdraw
Tx: 0h0m23s --- Rx: 0h0m27s ---
Last Connection Reset Reason:Rcv Notification
Notification Sent: Cease/CEASE Message
Notification Received: Cease/CEASE Message
Neighbor NLRI Negotiation:
Peer Negotiated IPV6 unicast capability
Peer configured for IPV6 unicast Routes
Neighbor ipv6 MPLS Label Capability Negotiation:
Neighbor AS4 Capability Negotiation:
Outbound Policy Group:
ID: 2, Use Count: 2
Update running at: 0.0.0.0/0
Last update time was 104 sec ago
Byte Sent: 158, Received: 0
Local host: 2001::2, Local Port: 8168
Remote host: 2001::1, Remote Port: 179
```

This example output gives summarized information about BGP4+ neighbors.

2. Enter the **show ipv6 bgp neighbors advertised-routes** command.

```
device# show ipv6 bgp neighbor 2001:db8::10 advertised-routes
       There are 7 routes advertised to neighbor 2001:db8::10
Status A:AGGREGATE B:BEST b:NOT-INSTALLED-BEST E:EBGP I:IBGP L:LOCAL
       Prefix             Next Hop        MED        LocPrf     Weight Status
1      fd80:122:122:122:101:101:0:122/128  2001:122:122::122
                                          0          100        101    BL
         AS_PATH:
2      fd80:122:122:122:103:103:0:122/128  2001:122:122::122
                                          0          100        103    BL
         AS_PATH:
3      fd80:122:122:122:105:105:0:122/128  2001:122:122::122
                                          0          100        105    BL
         AS_PATH:
4      131::1/128         2001:122:122::122
                                          1          100        32768  BL
         AS_PATH:
5      2001:122:131:125:131:1::/96  2001:3002::732
                                          1          100        0      BE
         AS_PATH: 65530
6      2001:abcd:1234:1234:1:2:1:0/112  2001:3002::733
                                          1          100        0      BE
         AS_PATH: 65530
7      2001:abcd:1234:1234:1:2:2:0/112  2001:3002::733
                                          1          100        0      BE
```

This example shows information about all the routes the BGP4+ networking device advertised to the neighbor.

3. Enter the **show ipv6 bgp neighbors last-packet-with-error** command.

```
device# show ipv6 bgp neighbor last-packet-with-error
Total number of BGP Neighbors: 67
1 IP Address: 153::2
    Last error:
    BGP4: 0 bytes hex dump of packet that contains error
```

This example shows information about the last packet that contained an error from any of a device's neighbors.

4. Enter the **show ipv6 bgp neighbors received-routes** command.

```
device# show ipv6 bgp neighbor 2001:db8::10 received-routes
There are 4 received routes from neighbor 2001:db8::10
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     2001:db8:2002::/64    2001:db8::10     0    100    0    BE
AS_PATH: 400
2    2001:db8:2003::/64    2001:db8::10     1    100    0    BE
AS_PATH: 400
3    2001:db8:2004::/64    2001:db8::10     1    100    0    BE
AS_PATH: 400
4    2001:db8:2005::/64    2001:db8::10     1    100    0    BE
AS_PATH: 400
```

This example lists all route information received in route updates from BGP4+ neighbors of the device since the soft-reconfiguration feature was enabled.

5. Enter the **show ipv6 bgp neighbors rib-out-routes** command.

```
device# show ipv6 bgp neighbors 2001:db8::10 rib-out-routes
       There are 150 RIB_out routes for neighbor 2001:db8::10
Status A:AGGREGATE B:BEST b:NOT-INSTALLED-BEST E:EBGP I:IBGP L:LOCAL
       Prefix              Next Hop        MED        LocPrf     Weight Status
1      fd80:122:122:122:101:101:0:122/128  ::
                                           0          100        101    BL
         AS_PATH:
2      fd80:122:122:122:103:103:0:122/128  ::
                                           0          100        103    BL
         AS_PATH:
3      fd80:122:122:122:105:105:0:122/128  ::
                                           0          100        105    BL
         AS_PATH:
4      131::1/128          ::              1          100        32768  BL
         AS_PATH:
5      2001:122:131:125:131:1::/96  2001:3002::732
                                           1          100        0      BE
         AS_PATH: 65530
6      2001:abcd:1234:1234:1:2:1:0/112  2001:3002::733
                                           1          100        0      BE
         AS_PATH: 65530
7      2001:abcd:1234:1234:1:2:2:0/112  2001:3002::733
                                           1          100        0      BE
         AS_PATH: 65530
```

This example shows information about BGP4+ outbound RIB routes.

Clearing BGP4+ dampened paths

BGP4+ suppressed routes can be reactivated using a CLI command.

The **show ipv6 bgp dampened-paths** command is entered to verify that there are BGP4+ dampened routes. The **clear ipv6 bgp dampening** command is entered to reactivate all suppressed BGP4+ routes. The **show ipv6 bgp dampened-paths** command is re-entered to verify that the suppressed BGP4+ routes have been reactivated.

1. Enter the **exit** command until you return to Privileged EXEC mode.

```
device(config)# exit
```

2. Enter the **show ipv6 bgp dampened-paths** command to display all BGP4+ dampened routes.

```
device# show ipv6 bgp dampened-paths

        Network               From                   Flaps     Since     Reuse     Path
*d      2001:db8:8::/45     2001:db8:1::1     1     0 :1 :14     0 :2 :20     100 1002 1000
*d      2001:db8:1::/48     2001:db8:1::1     1     0 :1 :14     0 :2 :20     100 1002 1000
*d      2001:db8:4::/46     2001:db8:1::1     1     0 :1 :14     0 :2 :20     100 1002 1000
*d      2001:db8:2::/47     2001:db8:1::1     1     0 :1 :14     0 :2 :20     100 1002 1000
*d      2001:db8:0:8000::/49     2001:db8:1::1     1     0 :1 :14     0 :2 :20     100 1002 1000
*d      2001:db8:17::/64    2001:db8:1::1     1     0 :1 :18     0 :2 :20     100
```

3. Enter the **clear ipv6 bgp dampening** command to reactivate all suppressed BGP4+ routes.

```
device# clear ipv6 bgp dampening
```

4. Enter the **show ipv6 bgp dampened-paths** command to verify that there are no BGP4+ dampened routes.

```
device# show ipv6 bgp dampened-paths
device#
```

The following example reactivates all suppressed BGP4+ routes and verifies that there are no suppressed routes.

```
device(config-bgp-router)# exit
device(config-rbridge-id-122)# exit
device(config)# exit
device# show ipv6 bgp dampened-paths
device# clear ipv6 bgp dampening
device# show ipv6 bgp dampened-paths
```

BGP EVPN

Overview of controllerless network virtualization with BGP EVPN

Layer 2 extension mechanisms using VXLAN rely on “flood and learn” mechanisms. These mechanisms are very inefficient, making MAC address convergence longer and resulting in unnecessary flooding.

Also, in a data center environment with VXLAN-based Layer 2 extension mechanisms, a Layer 2 domain and an associated subnet might exist across multiple racks and even across all racks in a data center site. With traditional underlay routing mechanisms, routed traffic destined to a VM or a host belonging to the subnet follows an inefficient path in the network, because the network infrastructure is aware only of the existence of the distributed Layer 3 subnet, but is not aware of the exact location of the hosts behind a leaf switch.

With Extreme BGP EVPN network virtualization, network virtualization is achieved through creation of a VXLAN-based overlay network. Extreme BGP EVPN network virtualization leverages BGP EVPN to provide a control plane for the virtual overlay network. BGP EVPN enables control-plane learning for end hosts behind remote VXLAN tunnel endpoints (VTEPs). This learning includes reachability for Layer 2 MAC addresses and Layer 3 host routes.

Some key features and benefits of Extreme BGP EVPN network virtualization are summarized as follows:

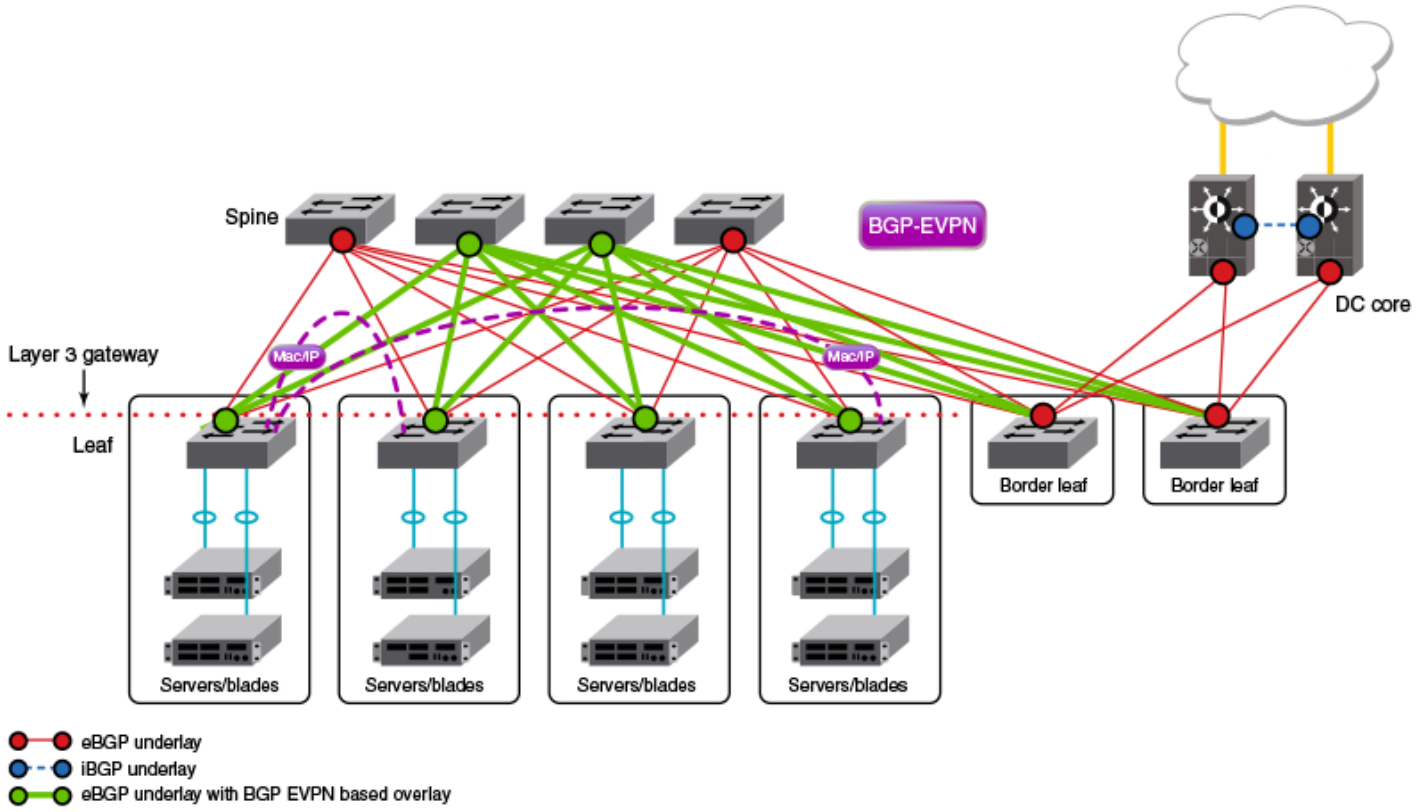
- Active-active vLAG pairs: node pairs for multiswitch port channel for dual homing of network endpoints are supported at the leaf. Both the switches in the vLAG pair participate in the BGP EVPN operations and are capable of actively forwarding traffic.
- Static anycast gateway: With static anycast gateway technology, each leaf is assigned the same default gateway IP and MAC addresses for all the connected subnets. This ensures that local traffic is terminated and routed at Layer 3 at the leaf. This also eliminates any suboptimal inefficiencies found with centralized gateways. All leaves are simultaneously active forwarders for all default traffic for which they are enabled. Also, because the static anycast gateway does not rely on any control plane protocol, it can scale to large deployments.
- Efficient VXLAN routing: With the existence of active-active leaf and the static anycast gateway, all traffic is routed and switched at the leaf. Routed traffic from the network endpoints is terminated in the leaf and is then encapsulated in VXLAN header to be sent to the remote site. Similarly, traffic from the remote leaf node is VXLAN-encapsulated and needs to be decapsulated and

routed to the destination. This VXLAN routing operation into and out of the tunnel on the leaf switches is enabled in the Extreme platform ASICs. VXLAN routing performed in a single pass is more efficient than with competitive ASICs.

- Data plane IP and MAC learning: With IP host routes and MAC addresses learned from the data plane and advertised with BGP EVPN, the leaf switches are aware of the reachability information for the hosts in the network. Any traffic destined to the hosts takes the most efficient route in the network.
- Layer 2 and Layer 3 multitenancy: BGP EVPN provides control plane for VRF routing as well as for Layer 2 VXLAN extension. BGP EVPN enables a multitenant infrastructure and extends it across the data center site to enable traffic isolation between the Layer 2 and Layer 3 domains, while providing efficient routing and switching between the tenant endpoints.
- Dynamic tunnel discovery: With BGP EVPN, the remote VTEPs are automatically discovered. The resulting VXLAN tunnels are also automatically created. This significantly reduces Operational Expense (OpEx) and eliminates errors in configuration.
- ARP/ND suppression: As the BGP EVPN EVI leaves discover remote IP and MAC addresses, they use this information to populate their local ARP tables. Using these entries, the leaf switches respond to any local ARP queries. This eliminates the need for flooding ARP requests in the network infrastructure.
- Conversational ARP/ND learning: Conversational ARP/ND reduces the number of cached ARP/ND entries by programming only active flows into the forwarding plane. This helps to optimize utilization of hardware resources. In many scenarios, there are software requirements for ARP and ND entries beyond the hardware capacity. Conversational ARP/ND limits storage-in-hardware to active ARP/ND entries; aged-out entries are deleted automatically.
- VM mobility support: If a VM moves behind a leaf switch, with data plane learning, the leaf switch discovers the VM and learns its addressing information. It advertises the reachability to its peers, and when the peers receive the updated information for the reachability of the VM, they update their forwarding tables accordingly. BGP EVPN-assisted VM mobility leads to faster convergence in the network.
- Simpler deployment: With multi-VRF routing protocols, one routing protocol session is required per VRF. With BGP EVPN, VRF routing and MAC address reachability information is propagated over the same BGP sessions as the underlay, with the addition of the L2VPN EVPN address family. This significantly reduces OpEx and eliminates errors in configuration.
- Open standards and interoperability: BGP EVPN is based on the open standard protocol and is interoperable with implementations from other vendors. This allows the BGP EVPN-based solution to fit seamlessly in a multivendor environment

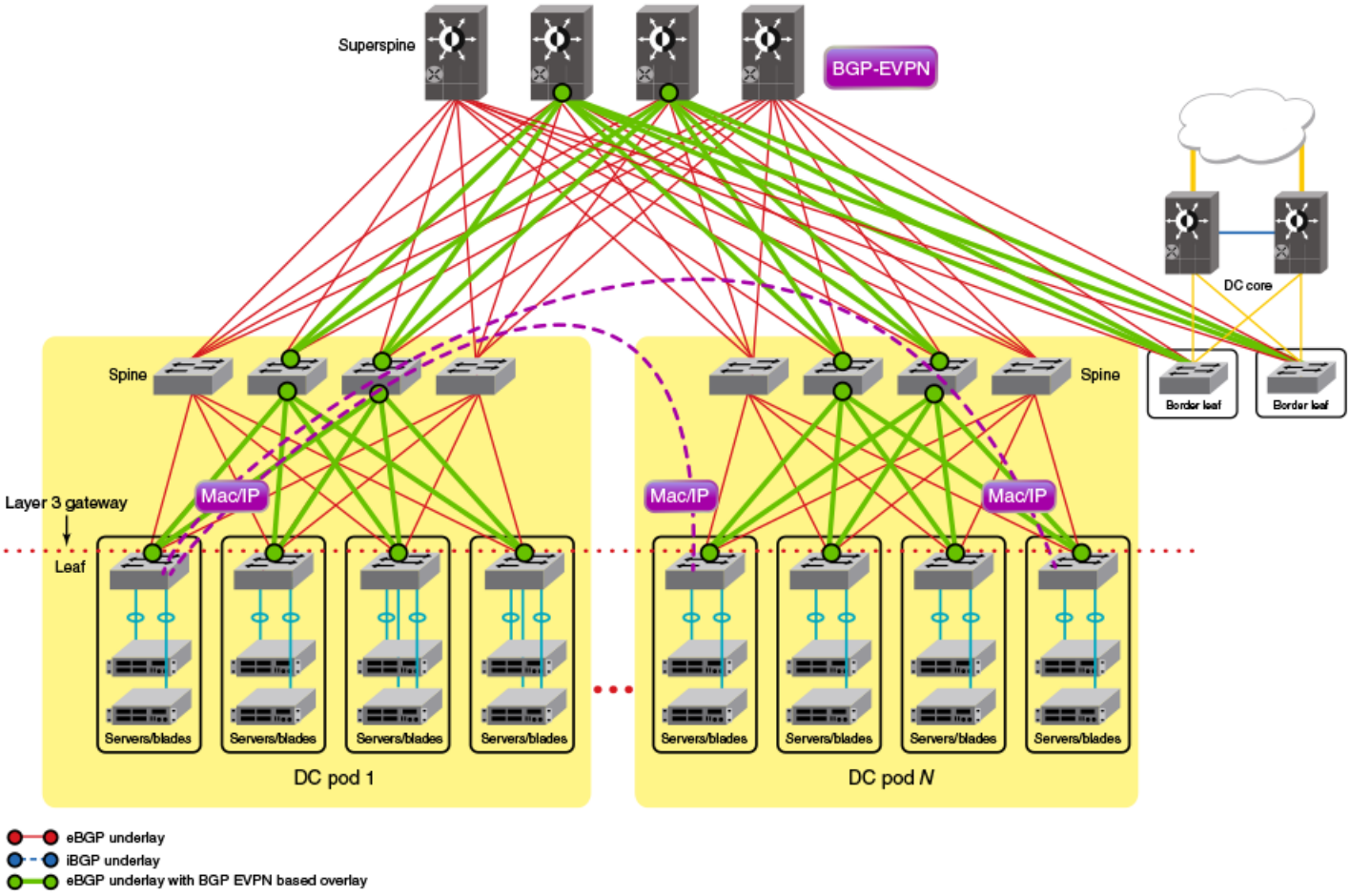
The following illustration depicts a 3-stage Clos topology that uses BGP EVPN. VXLAN tunnels are represented by dashed lines. This topology allows the provider to extend VLANs across racks with support from a Layer 3 underlay. Note that there is no controller, as VTEP autodiscovery is used. VTEPs are automatically discovered across the racks, and all MAC and IP address learning occurs automatically on the control plane. In this case, manual or controller intervention is not required. BGP EVPN can be extended to the border leaf nodes.

FIGURE 26 Overlay for a 3-stage Clos network without a controller



The following figure depicts a topology similar to that shown above, but with an optional optimized 5-stage folded Clos topology (superspine) for data center connectivity. Here the border leaf nodes connect directly to the superspine nodes. BGP EVPN can be extended to the border leaf nodes.

FIGURE 27 Overlay for an optimized 5-stage folded Clos topology without a controller



BGP-based underlay architecture supporting BGP EVPN

This section illustrates the basic BGP underlay architecture that is required to support the BGP EVPN overlay.

- eBGP-based BGP EVPN on page 290
- iBGP-based BGP EVPN on page 292

eBGP-based BGP EVPN

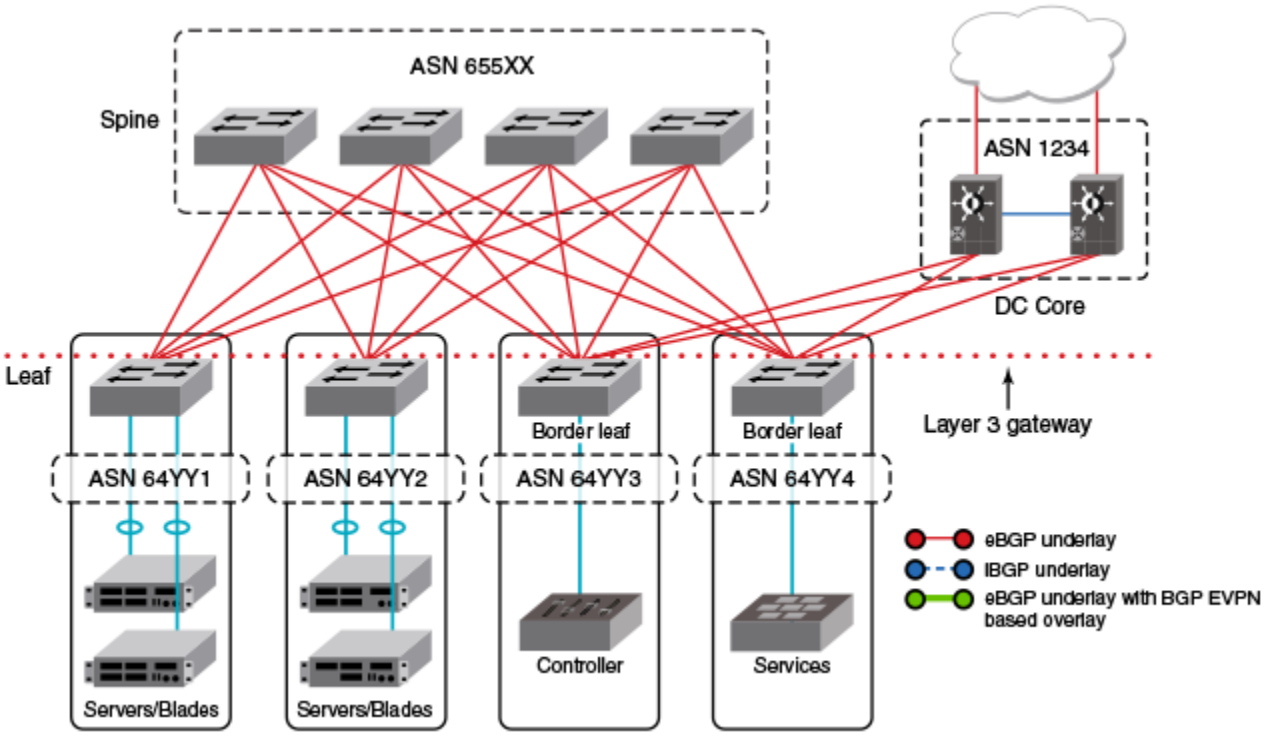
When eBGP is used for BGP EVPN in an IP Fabric, spine switches are configured to be in one autonomous system (AS). Each leaf switch or ToR pair is configured to be in a different AS from that of the spine switches, and advertises the routes by using local VTEP IP addresses as the next hop address.

For the BGP EVPN address family, spine switches are configured to advertise these routes to other eBGP peers, leaving the next-hop attribute unchanged.

NOTE
The spines are configured with separate VCS IDs.

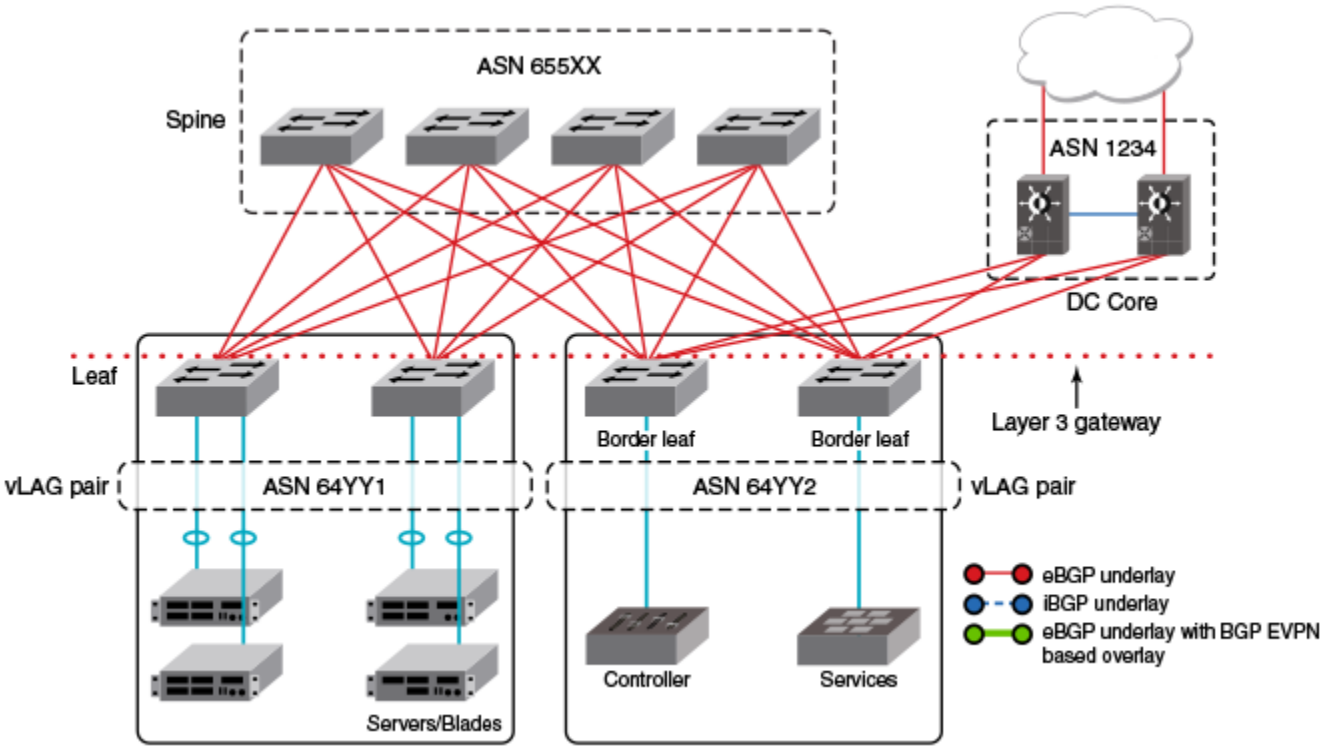
The following figure shows the eBGP underlay configuration in an IP Fabric where the spine switches are configured to be in one AS. Each leaf switch is configured to be in a separate AS.

FIGURE 28 eBGP underlay configuration with leaf switches in a separate autonomous system



The following figure shows the eBGP underlay configuration in an IP Fabric where the spine switches are configured to be in one AS. Leaf 1 and leaf 2 are configured to be in one AS, and leaf 3 and leaf 4 are configured to be in another AS.

FIGURE 29 eBGP underlay configuration with leaf switches in two separate autonomous systems



NOTE

When eBGP is used in conjunction with the IP unnumbered interfaces feature, it is important to ensure that sufficient Time To Live (TTL) values are available for hello messages to establish separate neighbor adjacencies on leaf switches in an IP Fabric. To do so, use the **neighbor ebgp-multihop** command and set the number of maximum hops to 2. Refer to the *Command Reference* for more information.

iBGP-based BGP EVPN

When iBGP is used for BGP EVPN in an IP Fabric, spine switches must be configured as route reflectors (RRs). This is because iBGP generally requires a switch to peer with every other device in the IP Fabric. When the spine switch is configured as an RR this situation is avoided.

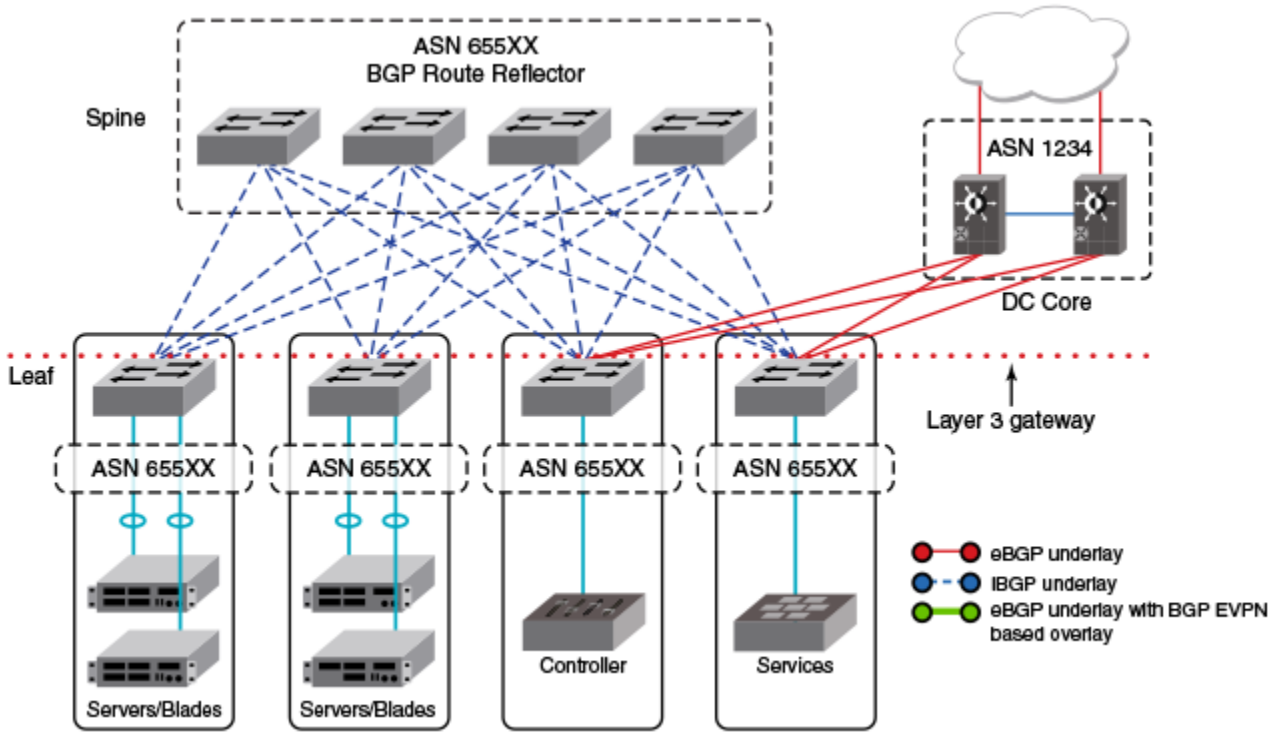
However, BGP RRs only reflect the best route. Only the single, best prefix is reflected to each leaf, which is configured as an RR client, even if multiple Equal-Cost Multipath (ECMP) routes exist.

NOTE

To enable faster convergence with ECMP routes, and ensure that a BGP RR sends multiple paths to the clients, the BGP add path feature must be configured in IPv4 or IPv6 address-family unicast configuration mode to advertise additional ECMP paths to the clients. To meet all IP Fabrics requirements when using iBGP for an IP Fabric, spine switches must support BGP RR and BGP add path. The entire IP Fabric can then be managed as a single ASN.

The following figure shows the iBGP underlay configuration for an IP Fabric. The entire IP Fabric is managed as a single ASN.

FIGURE 30 iBGP underlay configuration in an IP Fabric



BGP add path

The BGP add path feature provides the ability for multiple paths for the same prefix to be advertised without the new paths implicitly replacing the previous paths. Path diversity is achieved rather than path hiding.

When iBGP is used for BGP EVPN in an IP Fabric, spine switches are configured as route reflectors (RRs) so that a switch is not required to peer with every other device in the IP Fabric. Leaf switches are configured as RR clients.

For more information on BGP add path, refer to the BGP and BGP4+ chapters.

BGP EVPN NLRI

BGP EVPN distributes Network Layer Reachability Information (NLRI) for a network. BGP EVPN NLRI includes both Layer 2 and Layer 3 reachability information for end hosts residing in the network, and advertises both the MAC and IP addresses of the EVPN VXLAN end hosts.

The following table shows BGP EVPN support for NLRI.

TABLE 14 BGP EVPN support for NLRI

Support for NLRI	Description
Auto-discovery (AD) per ES	An ESI can be associated with more than one EVPN instance. When an interface with an associated ESI and EVPN instance comes online, an auto-discovery (AD) route per ES route is originated and installed in the corresponding EVPN instance. The ESI can be distinguished locally through the IP address of the peer.
MAC and MAC IP	The MAC and MAC-IP addresses of the EVPN VXLAN end hosts are advertised to peers, and the distribution of these addresses through BGP EVPN reduces unknown unicast flooding in the VXLAN.

TABLE 14 BGP EVPN support for NLRI (continued)

Support for NLRI	Description
Ethernet Tag Routes	Multicast Ethernet Tag routes advertise VLAN membership to peers, indicating the type of multicast tunnel on which flooded traffic can be received for a VLAN.
ES-Routes	Multihoming support is provided through the advertisement of BGP Ethernet Segment Routes (ES-Routes). An Ethernet Segment (ES) is the set of links connected to the same multihomed host. An Ethernet Segment Identifier (ESI) is associated with each Ethernet segment and advertised in the Ethernet Segment Route (ES-Route). For BGP EVPN, BGP initiates an ES-Route in the EVPN routing table and advertises it to EVPN neighbors, thus distributing NLRI for the network.
Inclusive Multicast Route (IMR)	BGP EVPN uses this BGP Type 0x3 Route Type to advertise multicast labels for multicast tunnel end-point discovery.
Prefix Routes	IPv4 and IPv6 prefix routes belonging to tenants (VRFs) are exchanged providing inter-subnet connectivity within the data center.

MAC address learning

Data plane (Layer 2) MAC address learning is enabled by default at the end points of the statically configured VXLAN tunnel..

Where BGP routing is used, MAC learning can be enabled by means of the **mac-learning protocol bgp** command. This delegates the responsibility for MAC learning to the BGP control-plane protocol. When VXLAN tunnel auto-discovery is enabled in BGP EVPN, default MAC learning is done by means of BGP.

For more information on MAC address learning, refer to Configuring MAC learning of Layer 2 extension site through BGP on page 301, as well as to the **mac-learning protocol bgp** command in the *Command Reference*.

EVPN instances

An EVPN instance (EVI) is an EVPN routing and forwarding instance spanning across the leaf nodes participating in that EVPN. EVIs are created using the **evpn-instance** command. Each EVI is identified by this configured name and is assigned an EVPN instance ID by the device.

Each EVI has a unique route distinguisher (RD) and one or more route targets (RT). RTs control the routes to be imported into and exported from the EVPN instance. An EVPN Routing table, containing information about the various routes associated with the EVI, is maintained for each EVI instance.

NOTE
Only one EVPN instance (EVI) is currently supported.

EVIs can be configured with the following features:

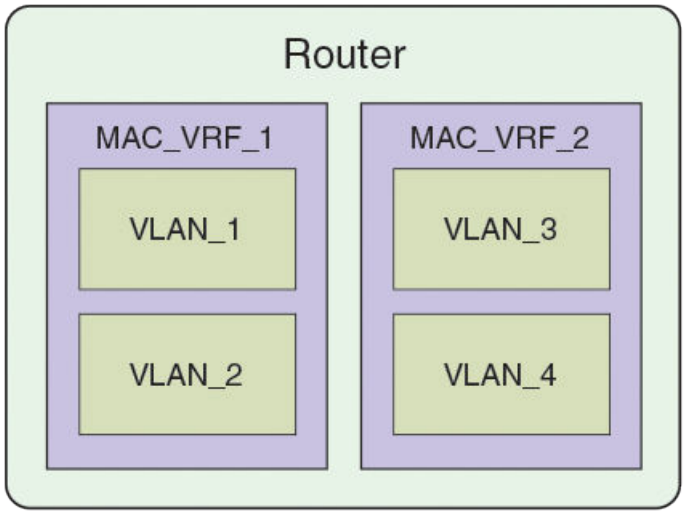
- Route distinguisher (RD): Unique route distinguishers are assigned for an EVI. This value is automatically derived globally using the **rd auto** command so that each EVI has an associated RD that is unique across the entire fabric. RDs can also be manually configured for a specific virtual network identifier (VNI) under an EVI using the **rd (VNI)** command.
- Route targets: The Route Target (RT) extended community attribute, enabling logical grouping of EVPN routes that can be imported or exported, can be specified for an EVI. These route targets can be globally derived automatically using the **route-target (EVPN)** command. The **ignore-as** option for the **route-target (EVPN)** command should be used in instances where the leaf nodes are under different autonomous systems and you want to import the routes from one leaf node to another. Route targets can also be manually configured for a specific virtual network identifier (VNI) under an EVI using the **route-target (VNI)** command. BGP EVPN uses these route targets to control the advertisement of EVPN routes.
- Virtual network identifiers (VNIs): VNIs can be added and removed for an EVI using the **vni add** and **vni remove** commands.

- Duplicate MAC detection timer: When the same host MAC address is learned by two different devices, continuous MAC moves are triggered by the traffic originating from these hosts. A duplicate MAC detection timer can be configured, using the **duplicate-mac-timer** command, to detect these continuous MAC moves. This timer specifies both a time interval and the maximum threshold of MAC moves that can occur within the configured time interval before the MAC address is treated as a duplicate address and further processing of updates for that MAC address are blocked.

For more information on the EVPN commands, refer to the *Extreme Network OS Command Reference*.

The following figure shows an RBridge with two configured EVIs, MAC_VRF_1 and MAC_VRF_2. Each EVI has a separate MAC-to-VLAN table.

FIGURE 31 RBridge with two configured EVIs



NOTE
Only one EVPN instance (EVI) is currently supported.

BGP L2VPN EVPN address family

The BGP L2VPN EVPN address-family configuration level provides access to commands that allow you to configure BGP EVPN. The BGP L2VPN EVPN address family uses components of BGP that are independent of the IPv4 and IPv6 unicast address families. Both iBGP and eBGP are supported.

The L2VPN EVPN address family supports the EVPN Subsequent Address Family Identifier (SAFI), an address qualifier that provides additional information about the Network Layer Reachability Information (NLRI) type for a given attribute.

The commands that you enter at this level apply only to the BGP L2VPN EVPN address family. BGP L2VPN EVPN address family capability can be negotiated along with the IPv4 or IPv6 address family. A separate BGP session is not required for the BGP EVPN address family.

To negotiate only the BGP L2VPN EVPN address family on an IPv4 neighbor, you must explicitly deactivate the IPv4 unicast address family using the **no neighbor activate** command.

To negotiate only the BGP L2VPN EVPN address family on an IPv6 neighbor, you must activate the neighbor only under the BGP L2VPN EVPN address family.

The following configuration options are allowed under BGP address-family L2VPN EVPN configuration mode:

```
device(config-bgp-evpn)# ?

Possible completions:

 client-to-client-reflection   Configure client to client route reflection
 graceful-restart              Enables the BGP graceful restart capability
 neighbor                      Specify a neighbor router
 retain                        Retain route targets
 vtep-discovery                Enable VTEP discovery
```

The following neighbor configuration options are allowed under BGP address-family L2VPN EVPN configuration mode:

```
device(config-bgp-evpn)# neighbor 10.1.1.1 ?

Possible completions:

  activate                 Allow exchange of route in the current family mode
  allowas-in               Disables the AS_PATH check of the routes learned
                           from the AS
  enable-peer-as-check     Disable routes advertise between peers in same AS
  maximum-prefix
  next-hop-unchanged       Next hop unchanged
  route-map                Apply route map
  route-reflector-client   Configure a neighbor as Route Reflector client
  send-community           Send community attribute to this neighbor
```

Automatic VXLAN tunnel endpoint discovery

VXLAN tunnel endpoint (VTEP) IP addresses are carried in every BGP EVPN route so that the leaf device receiving the BGP EVPN updates triggers VXLAN tunnel creation using this remote VTEP IP address. Remote VTEP discovery is enabled by default for BGP EVPN when the BGP L2VPN EVPN address family is enabled.

BGP devices can determine which VLANs are common between two devices and extend those VLANs over the VXLAN tunnel between the two devices. VTEP IP address is carried in BGP EVPN updates in next-hop network address field of the MP_REACH_NLRI attributes. BGP deletes VXLAN tunnels associated with remote VTEP when all of the routes from remote VTEP are withdrawn. If you want to configure a tunnel explicitly (statically) or in an NSX Controller deployment, the BGP Automatic VTEP feature should be disabled by means of the **no vtep-discovery** command.

BGP next hop unchanged

The BGP next hop unchanged feature is supported for the L2VPN EVPN address family, allowing BGP to send updates to eBGP peers with the next hop in the MP_REACH_NLRI attribute unchanged.

By default, eBGP BGP speakers change the next hop while sending the updates to eBGP neighbors. The BGP next-hop-unchanged feature overrides this behavior so that the next hop address remains unchanged while updates are sent to eBGP peers. The BGP speaker is forced to retain the next hop address in the BGP updates received from neighbors. BGP next-hop-unchanged should be configured on the spine switch for each EVPN neighbor if the leaf switch is an eBGP peer. BGP next-hop-unchanged should be configured on the leaf switch for each EVPN neighbor if the spine switch is an eBGP peer. This means that eBGP BGP speakers will not change the next hop while sending updates to eBGP neighbors in an IP Fabric. Therefore, by configuring BGP next-hop-unchanged under the BGP L2VPN EVPN address family, changes to the next hop address in BGP EVPN updates are prevented.

NOTE

BGP next-hop-unchanged should be configured on leaf nodes and on spine nodes for all types of BGP deployments.

BGP retain route target

The BGP retain route target feature allows a spine node to accept all route targets (RTs).

Because Spine switches do not have EVPN instances (EVIs) and VRFs configured, BGP EVPN routes are filtered as otherwise none of the RTs in the routes match the local route-target import configuration. When spine switches are configured as RRs, or establish eBGP peering with the leaf nodes, the BGP retain route target feature should be enabled so that the Spine switches do not do route-target filtering. This means that all route targets are retained and the route-target attributes in the EVPN routes are not modified or removed.

BGP retain route target is supported for the BGP L2VPN EVPN address family only and is used in BGP EVPN configurations. It is not supported for the IPv4 and IPv6 unicast address families.

RIBout AS path check

The RIBout AS path check feature enables the outbound AS_PATH check function so that a BGP sender speaker does not send routes with an AS path that contains the ASN of the receiving speaker. If the remote AS of the neighbor appears in the AS path segment of the NLRI, the NLRI is not added to RIBOut of the peer.

When the AS path check is enforced for the sender using the **neighbor enable-peer-as-check** command, it saves the amount of RIBout memory used to store routes that would otherwise be stored in sender's RIBout, advertised to the peer and be discarded by the receiver, if the **neighbor allowas-in** command is not configured. Convergence time is also improved.

BGP legacy features supported for the L2VPN EVPN address family

The following BGP features, already supported for IPv4 and IPv6 unicast address families, are supported for the BGP L2VPN EVPN address family and used in BGP EVPN configurations:

- BGP extended community: The BGP extended community feature filters routes based on a regular expression specified when a route has multiple community values in it. The BGP extended community feature is supported for the L2VPN EVPN address family for both spine and leaf nodes. When a spine or leaf node receives BGP EVPN routes from other spine nodes, it checks the route-target extended community attribute of the routes. If the route-target is the same as the import target of the given EVPN instance on the local leaf node, the leaf node adds the route to the EVPN routing table. If the spine node is configured with the BGP retain route target feature, this checking is bypassed in the spine. If a static MAC address is redistributed to BGP, it is advertised with the "Sticky MAC" extended community attribute. The next hop router MAC address for prefix routes is advertised as the default gateway extended community attribute. The same check is performed for prefix routes; however the comparison is made against the RT configured for VRFs rather than EVIs.
- BGP graceful restart: BGP graceful restart (GR) can be configured for the L2VPN EVPN address family. When GR capability is negotiated, neighboring devices participate in the restart, helping to ensure that no route and topology changes occur in the network for the duration of the restart. GR helps retain the peer routes when a restart occurs during HA failover. When the GR feature is enabled for the L2VPN EVPN address family, existing sessions are not affected and require neighbor reset to negotiate the GR capability.
- BGP peer groups: BGP peer groups can be configured for the L2VPN EVPN address family so that neighbors with the same attributes and parameters can be grouped together.
- BGP route reflection: The BGP route reflection feature is supported for the L2VPN EVPN address family. When iBGP is used for BGP EVPN in an IP Fabric, spine switches are configured as route reflectors (RRs) and leaf switches are configured as route

reflector clients. You can configure a leaf switch as a route reflector client from the spine switch using the **neighbor route-reflector-client** command. Each leaf switch should be configured so that they all belong in the same cluster.

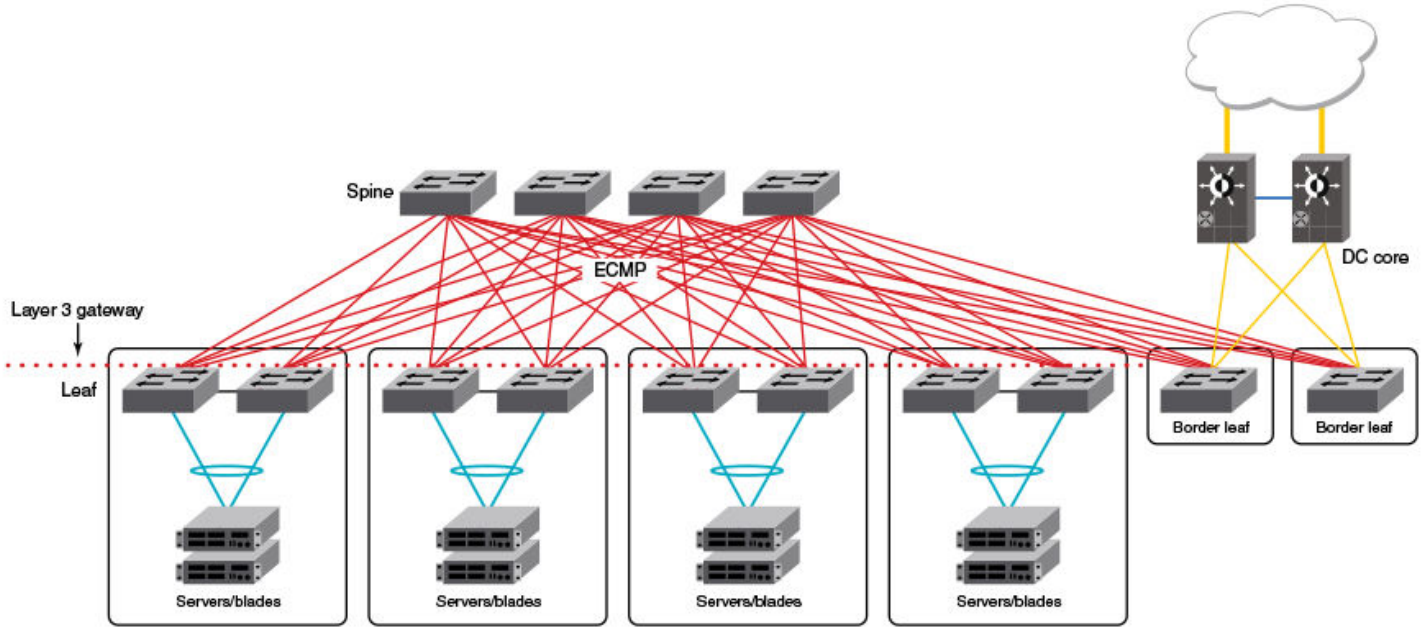
- Client-to-client-reflection: The BGP client-to-client reflection feature is supported for the L2VPN EVPN address family. For iBGP, if the leaf switches are configured as route reflector clients and are connected in a full iBGP mesh, you can disable client-to-client reflection on the spine switch which is configured as an RR. By default, in an IP Fabric, the clients of a route reflector are not required to be fully meshed and the routes from a client are reflected to other clients.
- Disabling the BGP AS_PATH check: A device can be configured so that the AS_PATH check function for routes learned from a specific location is disabled, and routes that contain the recipient BGP speaker's AS number are not rejected. When eBGP is configured for BGP EVPN in an IP Fabric, the AS_PATH check function can be disabled on a leaf switch so that route updates from the spine layer are not discarded by the leaf.

Multihoming

Multihoming is an optional feature that supports high availability.

When servers must be multihomed (specifically, dual homed; refer to the following figure) to more than one leaf switch in an IP Fabric (as for optional HA support), the leaf switches must be configured as a vLAG pair. In this case, (1) RBridge IDs must be unique for each node in the pair, in order to form the local VCS Fabric; (2) the same loopback addresses must be configured on each node to support distributed VTEP if required; (3) both nodes must have the same VCS ID; and (4) a unique router ID (configured by means of the **ip router-id** command) must be configured on each RBridge, so that each RBridge can associate unique route identifiers.

FIGURE 32 A 3-stage folded Clos topology with redundant servers or blades



NOTE

For the details of vLAG configuration, refer to the "Link Aggregation" chapter in the *Extreme Network OS Layer 2 Switching Configuration Guide*.

Note the following considerations and limitations for this scenario:

- A pair of switches to which other servers create dual-homed connections forms a vLAG pair.
- A VXLAN tunnel is not established between the nodes in a vLAG pair.
- Distributed VXLAN gateway functionality is used to extend VXLAN tunnels to other leaf nodes.
- The Static Anycast Gateway and VRRP-E features are options for providing redundant gateway functionality.
- If two leaf nodes are connected by means of an external Layer 2 switch, this can result in a multihoming situation for hosts connected to that switch. This topology is not supported.

When two leaf nodes such as those shown in the previous figure operate in a vLAG pair, they form independent BGP sessions with other routers, including other nodes in the same IP Fabric. It is recommended that the same **local-as** value be configured for all switches belonging to the same vLAG pair. For leaf switches to accept BGP updates, **neighbor allowas-in** must be configured, to disable the AS_PATH check function from rejecting routes that contain the recipient BGP speaker's AS number (ASN).

As noted above, all switches in the vLAG pair must be configured with the same loopback VTEP IP address. This ensures that VTEP discovery does not form a VXLAN tunnel between the two nodes. (The tunnel source and destination IP addresses are the same.) This supports redundancy for the tunnels, as provided in the previous release.

Ethernet Segment Identifiers for BGP routing

An Ethernet Segment is a set of Ethernet links that connect a multihomed device to a BGP router. Ethernet Segments are assigned a unique identifier, referred to as an Ethernet Segment Identifier (ESI). ESIs can be configured on port-channel interfaces.

Ethernet links that connect single-homed devices are not required to have an ESI value associated with them. For BGP EVPN, each BGP device advertises an Ethernet Segment Route (ES-Route) informing other BGP devices that it is connected to that ES. The other BGP devices can then detect if they are connected to the same ES. The user can use the **esi** command in port-channel configuration mode to configure the port-channel to derive the ESI value automatically by means of Link Aggregation Control Protocol (LACP). ESI value can be manually also configured on the port channel. This is needed for static port channels where LACP is not involved.

Multi-VRF for BGP EVPN

Multi-VRF must be configured in an IP Fabric to support asymmetric or symmetric routing. The link between the leaf and spine switches must be configured in the default VRF. For a nondefault VRF, traffic is routed at the leaf switch and is forwarded to a VXLAN tunnel.

For asymmetric routing, all VLANs are configured on every leaf switch and enabled for IP routing and forwarding. Traffic flows through different routes in different directions. This severely affects scaling with each switch carrying the MAC and MAC-IP routes for hosts that are not necessarily locally connected.

For symmetric routing, a VLAN is configured only at the leaf switch where a local host exists. A unique, common Layer 3 virtual network identifier (VNI) number is generated for all leaf nodes for the VRF instance. The ingress leaf switch performs a Layer 3 lookup and routes the packets towards the remote leaf switch over the common Layer 3 VNI. The inner MAC destination address (DA) of the packet is rewritten to that of the gateway MAC address advertised by the remote leaf switch and the packet is then encapsulated in a VXLAN tunnel and sent to the remote leaf switch. The remote leaf switch then terminates the VXLAN tunnel. Because the MAC DA of the inner packet is now that of the gateway MAC address advertised by the leaf switch, it performs Layer 3 lookup and the packet is routed to the locally attached host.

VRFs can be configured with the following features:

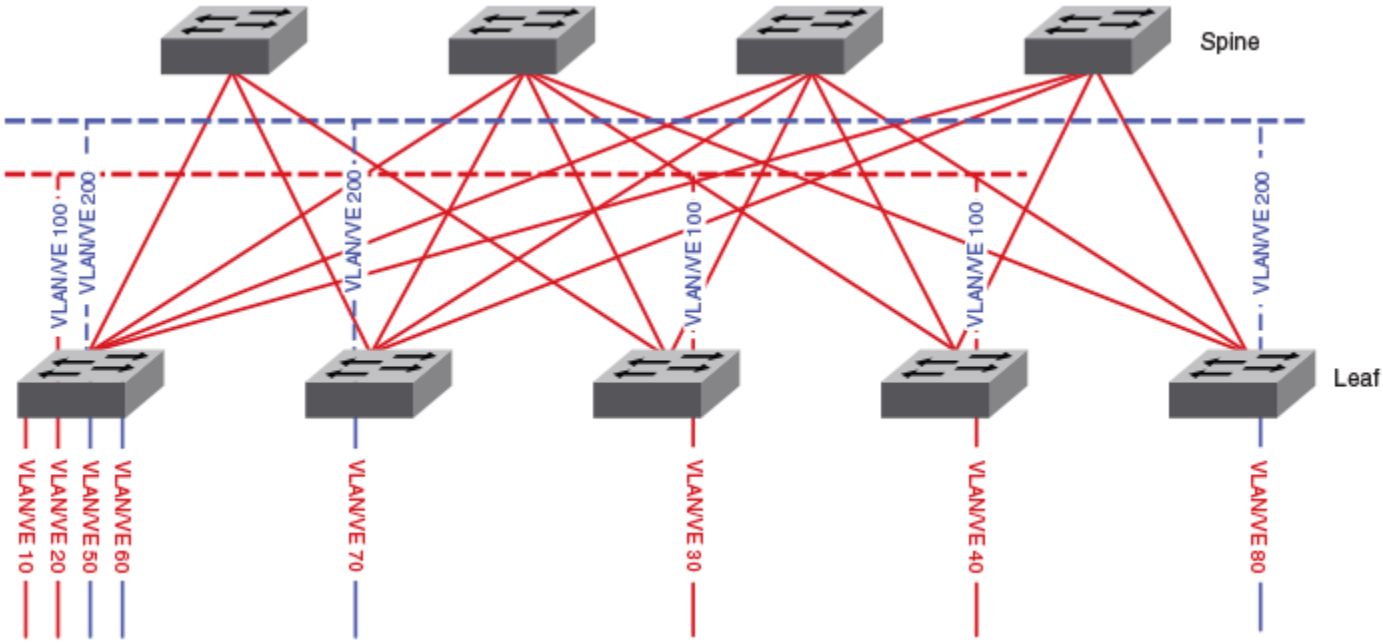
- Virtual network identifiers (VNIs): Layer 3 VNIs can be added and removed for a VRF instance. The VNI configuration under a VRF instance tells BGP to include the Layer 3 VNI and its associated MAC address in EVPN updates.

- Route distinguisher (RD): Unique route distinguishers are assigned for a VRF instance. RDs are manually configured for a VRF instance using the **rd (VRF)** command.
- Route targets: The route target (RT) extended community attribute, enabling logical grouping of EVPN routes that can be imported or exported, can be specified for a VRF instance. Route targets are manually configured for a specific VRF instance using the **route-target (VRF)** command. BGP EVPN uses these route targets to control the exchange of EVPN routes.
- Route filters: Filtering based on route-map policies can be added to the EVPN IPv4 and IPv6 prefix routes that are imported to or exported from VRFs. Refer to "BGP VRF route filters" in the "BGP4" and "BGP4+" chapters in this document.

For more information on the EVPN commands, refer to the *Command Reference*.

The following figure illustrates Multi-VRF topology using BGP EVPN. In this figure VRF red has VLAN 10, 20, 30, and 40. VLAN 100 is the Layer 3 VNI for VRF red. VRF blue has VLAN 50, 60, 70, and 80. VLAN 200 is the Layer 3 VNI for VRF blue.

FIGURE 33 Multi-VRF for BGP EVPN



The leaking of IPv4 and IPv6 prefix routes across VRFs is allowed using the BGP-EVPN control plane across leaf nodes. If you configure route-targets of the source VRF as import route-targets, routes originated at a leaf node in a given VRF can be imported into one or more VRFs by a node. Routes from a given VRF, including MAC IP (ARP) routes that were converted to /32 prefix routes and installed into the RIB, can be leaked into multiple VRFs. Refer to the **route-target (VRF)** command in the *Command Reference* for more information.

Configuring BGP EVPN

BGP EVPN can be used to implement the overlay network for an IP Fabric using the tasks outlined in this section. Refer to the *Extreme Network OS IP Fabrics Configuration Guide* for more information on IP Fabrics.

Configuring MAC learning of Layer 2 extension site through BGP

The user can choose the way MAC learning is achieved for a Layer 2 extension tunnel.

BGP routing must be configured.

Data plane (Layer 2) MAC address learning is enabled by default at the remote site. However, this leads to scalability issues. Where BGP routing is used, do the following to enable MAC learning by means of BGP instead. This delegates the responsibility for MAC learning on a tunnel to the Layer 3 control-plane protocol, such as BGP EVPN.

1. Enter VXLAN overlay-gateway site configuration mode.

```
device# configure terminal
device(config)# overlay-gateway gateway1
device(config-overlay-gw-gateway1)# site mysite
device(config-overlay-gw-gateway1-site-mysite)#
```

2. Enter the **mac-learning protocol bgp** command.

```
device(config-overlay-gw-gateway1-site-mysite)# mac-learning protocol bgp
```

3. To disable BGP MAC learning and return to the default of Layer 2 learning, use the **no mac-learning protocol bgp** command.

```
device(config-overlay-gw-gateway1-site-mysite)# no mac-learning protocol bgp
```

(Optional) Configuring a BGP EVPN instance

A BGP EVPN instance (EVI) can be configured and various commands can be executed in EVPN instance configuration mode, as shown in the following configuration example.

BGP EVPN is not essential to the configuration of an IP Fabric, and is one among a variety of overlay solutions.

1. Enter the **configure terminal** command to access global configuration mode.

```
device# configure terminal
```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

```
device(config)# rbridge-id 1
```

3. Enter the **evpn-instance** command and specify a name to configure an EVPN instance and enter EVPN instance configuration mode.

```
device(config-rbridge-id-1)# evpn-instance myinstance
```

4. Enter the **rd auto** command to enable auto-generation of the RD value.

```
device(config-evpn-instance-myinstance)# rd auto
```

5. Enter the **route-target** command using the **both** and **auto** parameters to configure auto-generation of the import and export route-target community attributes globally.

```
device(config-evpn-instance-myinstance)# route-target both auto
```

6. Enter the **vni** command with the **add** *value* parameter to add VLANs to the EVI.

```
device(config-evpn-instance-myinstance)# vni add 1-100
```

   Ensure that only local VNIs are added, supporting local VLANs. With symmetric integrated routing and bridging (IRB), if a remote VNI is added that is not on the local switch, that VNI is treated as being used for Layer 2 extension and ARP entries are not programmed in hardware, resulting in potential performance problems. For example, if switch Switch-1 has only VNI 100 and another switch, Switch-2, has VNI 200, you would execute the**vlan add 100** command on Switch-1 and the **no vlan add 200** command on Switch-2.

7. (Optional) Enter the **duplicate-mac-timer** *count* command with the **max-count** *interval* parameter to set the duplicate MAC detection timer interval and the maximum count.

```
device(config-evpn-instance-myinstance)# duplicate-mac-timer 22 max-count 5
```

The following example configures the EVPN instance "myinstance" so that the RD value is generated automatically. The import and export route-target community attributes are also generated automatically. VLANs are added to the EVI. The duplicate MAC detection timer is set to 22 seconds and the number of times a MAC move can be detected in the configured 22 second interval before the MAC is suppressed is set to 5.

```
device# configure terminal
device(config)# rbridge-id 1
device(config-rbridge-id-1)# evpn-instance myinstance
device(config-evpn-instance-myinstance)# rd auto
device(config-evpn-instance-myinstance)# route-target both auto
device(config-evpn-instance-myinstance)# vlan add 1-100
device(config-evpn-instance-myinstance)# duplicate-mac-timer 22 max-count 5
```

Configuring interoperability with other vendors

When BGP EVPN is used, the automatic mapping of VLANs to VNIs may not work with all vendors.

Some vendors do not use the range of available VNIs (1 through 8191) that Extreme does. As a result of the automatic mapping process that is initiated by the **map vni auto** command in overlay-gateway configuration mode (entered by means of the **overlay-gateway** command), a vendor's VLANs that do not conform to the Extreme extended VLAN range must be remapped manually to their respective VNIs.

In addition, the route target (RT) and route distinguisher (RD) values are set automatically. Because such route targets use AS numbers in the administrator field, routes cannot be exchanged between leaf nodes belonging to different AS numbers. In such cases manual adjustments are required, as illustrated in this task.

1. From global configuration mode, enter overlay-gateway configuration mode and specify an overlay gateway.

```
device(config)# overlay-gateway gateway1
device(config-overlay-gw-gateway1)#
```

2. In overlay-gateway configuration mode, do the following.
   a) Use the **map vlan** command to map a VLAN to a VNI manually, as in the following example.

```
device(config-overlay-gw-gatweway1)# map vlan 100 vni 10100
```

   b) Repeat Step 2a as appropriate for additional manual mappings.

3. In RBridge ID configuration mode, enter EVPN configuration mode.

```
device# configure terminal
device(config)# rbridge-id 1
device(config-rbridge-id-1)# evpn-instance myinstance
device(config-evpn-instance-myinstance)#
```

4. In EVPN instance configuration mode, use the **route-target (EVPN)** command to specify auto-generation of the import and export route-target community attributes and ignore the ASN, and the **rd auto (EVPN)** command to enable auto-generation of a route distinguisher (RD) for an EVPN instance.

```
device(config-evpn-instance-myinstance)# route-target both auto ignore-as
device(config-evpn-instance-myinstance)# rd auto
```

5. In EVPN instance configuration mode, configure the VPN route distinguisher (RD) manually.
   a) Enter the **vni (EVPN)** command to specify the VNI and enter VNI instance configuration mode, where you use the **rd (VNI)** command and the **route-target (VNI)** command to specify manually the RD and RT, respectively.

```
device(config-evpn-instance-myinstance)# vni 10100
device(evpn-vni-10100)# rd 100:100
device(evpn-vni-10100)# route-target import 1:1
device(evpn-vni-10100)# route-target export 1:1
```

   b) Repeat Step 6a for all RDs and RTs that need to be mapped manually.

Enabling the BGP L2VPN EVPN address family

In order to configure BGP EVPN sessions, the BGP L2VPN EVPN address family must be configured.

1. Enter the **configure terminal** command to access global configuration mode.

```
device# configure terminal
```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

```
device(config)# rbridge-id 1
```

3. Enter the **router bgp** command to enable BGP routing.

```
device(config-rbridge-id-1)# router bgp
```

4. Enter the **neighbor** *ipv4* **remote-as** command to specify the ASN in which the remote neighbor resides.

```
device(config-bgp-router)# neighbor 10.1.1.1 remote-as 2
```

5. Enter the **address-family l2vpn evpn** command to access BGP address family L2VPN EVPN configuration mode.

```
device(config-bgp-router)# address-family l2vpn evpn
```

The following example configures accesses BGP address family L2VPN EVPN configuration mode so that BGP EVPN can be configured.

```
device# configure terminal
device(config)# rbridge-id 1
device(config-rbridge-id-1)# router bgp
device(config-bgp-router)# neighbor 10.1.1.1 remote-as 2
device(config-bgp-router)# address-family l2vpn evpn
device(config-bgp-evpn)#
```

Disabling automatic VXLAN tunnel endpoint discovery by BGP

Automatic VXLAN tunnel endpoint (VTEP) discovery by BGP is enabled by default and can be disabled by means of the following procedure.

1. Enter the **configure terminal** command to access global configuration mode.

```
device# configure terminal
```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

```
device(config)# rbridge-id 1
```

3. Enter the **router bgp** command to enable BGP routing.

```
device(config-rbridge-id-1)# router bgp
```

4. Enter the **address-family l2vpn evpn** command to enter L2VPN EVPN address family configuration mode.

```
device(config-bgp-router)# address-family l2vpn evpn
```

5. Enter the **no vtep-discovery** command to disable automatic VTEP discovery by BGP.

```
device(config-bgp-evpn)# no vtep-discovery
```

The following example disables automatic VTEP discovery by BGP.

```
device# configure terminal
device(config)# rbridge-id 1
device(config-rbridge-id-1)# router bgp
device(config-bgp-router)# address-family l2vpn evpn
device(config-bgp-evpn)# no vtep-discovery
```

Configuring BGP next hop unchanged

BGP next hop unchanged can be configured for the L2VPN EVPN address family, allowing BGP to send updates to eBGP peers with the next hop attribute unchanged. The **neighbor next-hop-unchanged** command should be configured for the Spine switch for each EVPN neighbor if the leaf switch is an eBGP peer. The **neighbor next-hop-unchanged** command should be configured for the leaf switch for each EVPN neighbor if the spine switch is an eBGP peer.

1. Enter the **configure terminal** command to access global configuration mode.

```
device# configure terminal
```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

```
device(config)# rbridge-id 1
```

3. Enter the **router bgp** command to enable BGP routing.

```
device(config-rbridge-id-1)# router bgp
```

4. Enter the **local-as** command to configure the autonomous system number (ASN) in which your device resides.

```
device(config-bgp-router)# local-as 1000
```

5. Enter the **neighbor** *ipv4* **remote-as** command to specify the ASN in which the remote neighbor resides.

```
device(config-bgp-router)# neighbor 10.1.1.1 remote-as 2
```

6. Enter the **address-family l2vpn evpn** command to enter L2VPN EVPN address family configuration mode.

```
device(config-bgp-router)# address-family l2vpn evpn
```

7. Enter the **neighbor** *ip address* **activate** command to establish a BGP EVPN session with the eBGP neighbor.

```
device(config-bgp-evpn)# neighbor 10.1.1.1 activate
```

8. Enter the **neighbor** *ip address* **neighbor next-hop-unchanged** command to configure BGP next hop unchanged.

```
device(config-bgp-evpn)# neighbor 10.1.1.1 next-hop-unchanged
```

The following example establishes a BGP EVPN session with an eBGP neighbor and configures BGP next hop unchanged so that updates can be sent to the neighbor with the next hop attribute unchanged.

```
device# configure terminal
device(config)# rbridge-id 1
device(config-rbridge-id-1)# router bgp
device(config-bgp-router)# local-as 1000
device(config-bgp-router)# neighbor 10.1.1.1 remote-as 2
device(config-bgp-router)# address-family l2vpn evpn
device(config-bgp-evpn)# neighbor 10.1.1.1 activate
device(config-bgp-evpn)# neighbor 10.1.1.1 next-hop-unchanged
```

Configuring BGP retain route target

BGP retain route target can be configured for the L2VPN EVPN address family so that a route reflector (RR) accepts all route targets (RTs). For iBGP, enable this feature on a spine switch so that route-target filtering is not performed and all route targets are retained and route-target attributes in the EVPN routes are not modified or removed.

1. Enter the **configure terminal** command to access global configuration mode.

```
device# configure terminal
```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

```
device(config)# rbridge-id 1
```

3. Enter the **router bgp** command to enable BGP routing.

```
device(config-rbridge-id-1)# router bgp
```

4. Enter the **local-as** command to configure the autonomous system number (ASN) in which your device resides.

```
device(config-bgp-router)# local-as 1000
```

5. Enter the **neighbor** *ipv4* **remote-as** command to specify the ASN in which the remote neighbor resides.

```
device(config-bgp-router)# neighbor 10.1.1.1 remote-as 2
```

6. Enter the **address-family l2vpn evpn** command to enter L2VPN EVPN address family configuration mode.

```
device(config-bgp-router)# address-family l2vpn evpn
```

7. Enter the **retain route-target all** command to configure the RR to accept all route targets RTs.

```
device(config-bgp-evpn)# retain route-target all
```

The following example configures a RR to accept all RTs.

```
device# configure terminal
device(config)# rbridge-id 1
device(config-rbridge-id-1)# router bgp
device(config-bgp-router)# local-as 1000
device(config-bgp-router)# neighbor 10.1.1.1 remote-as 2
device(config-bgp-router)# address-family l2vpn evpn
device(config-bgp-evpn)# retain route-target all
```

Applying a BGP extended community filter for the L2VPN EVPN address family

A BGP extended community filter can be applied for the L2VPN EVPN address family.

BGP communities must already be defined. For more information on defining BGP communities, refer to the “BGP4” and “BGP4+” chapters in the *Extreme Network OS Layer 3 Routing Configuration Guide*.

1. Enter the **configure terminal** command to access global configuration mode.

   ```
   device# configure terminal
   ```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

   ```
   device(config)# rbridge-id 1
   ```

3. Enter the **ip extcommunity-list** command and specify a number to set a BGP extended community filter.

   ```
   device(config-rbridge-id-1)# ip extcommunity-list 1 permit rt 123:2
   ```

4. Enter the **router bgp** command to enable BGP routing.

   ```
   device(config-rbridge-id-1)# router bgp
   ```

5. Enter the **local-as** command to configure the autonomous system number (ASN) in which your device resides.

   ```
   device(config-bgp-router)# local-as 1000
   ```

6. Enter the **neighbor** *ip-address* **remote-as** command to specify the ASN in which the remote neighbor resides.

   ```
   device(config-bgp-router)# neighbor 10.1.2.3 remote-as 1001
   ```

7. Enter the **address-family l2vpn evpn** command to enter L2VPN EVPN address family configuration mode.

   ```
   device(config-bgp-router)# address-family l2vpn evpn
   ```

8. Enter the **neighbor** *ip-address* **activate** command to enable the exchange of information with the neighbor.

   ```
   device(config-bgp-evpn)# neighbor 10.1.2.3 activate
   ```

9. Enter the **neighbor** *ip-address* **route-map** command and specify the **in** keyword to apply a route map to incoming routes.

   ```
   device(config-bgp-evpn)# neighbor 10.1.2.3 route-map in extComRmapt
   ```

10. Enter the **neighbor** *ip-address* **route-map** command and specify the **out** keyword to apply a route map to outgoing routes.

    ```
    device(config-bgp-evpn)# neighbor 10.1.2.3 route-map out sendExtComRmap
    ```

11. Enter the **neighbor** *ip-address* **send-community** command and specify the **both** keyword to enable the sending of standard and extended attributes in updates to the specified BGP neighbor.

```
device(config-bgp-evpn)# neighbor 10.1.2.3 send-community both
```

The following example applies a BGP extended community filter for the L2VPN EVPN address family. The steps for configuring a route map are not included in this example.

```
device# configure terminal
device(config)# rbridge-id 1
device(config-rbridge-id-1)# ip extcommunity-list 1 permit rt 123:2
device(config-rbridge-id-1)# router bgp
device(config-bgp-router)# local-as 1000
device(config-bgp-router)# neighbor 10.1.2.3 remote-as 1001
device(config-bgp-router)# address-family l2vpn evpn
device(config-bgp-evpn)# neighbor 10.1.2.3 activate
device(config-bgp-evpn)# neighbor 10.1.2.3 route-map in extComRmapt
device(config-bgp-evpn)# neighbor 10.1.2.3 route-map out sendExtComRmap
device(config-bgp-evpn)# neighbor 10.1.2.3 send-community both
```

Configuring BGP graceful restart for the L2VPN EVPN address family

Graceful restart can be configured for BGP EVPN in the L2VPN EVPN address family configuration mode, helping to retain peer routes and ensure that no route and topology changes occur in the network for the duration of a restart.

1. Enter the **configure terminal** command to access global configuration mode.

```
device# configure terminal
```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

```
device(config)# rbridge-id 1
```

3. Enter the **router bgp** command to enable BGP routing.

```
device(config-rbridge-id-1)# router bgp
```

4. Enter the **local-as** command to configure the autonomous system number (ASN) in which your device resides.

```
device(config-bgp-router)# local-as 1000
```

5. Enter the **neighbor** *ip address* **remote-as** command to specify the ASN in which the neighbor resides.

```
device(config-bgp-router)# neighbor 2001:db8:93e8:cc00::1 remote-as 2
```

6. Enter the **address-family l2vpn evpn** command to enter L2VPN EVPN address family configuration mode.

```
device(config-bgp-router)# address-family l2vpn evpn
```

7. Enter the **graceful-restart** command to enable the graceful restart feature.

```
device(config-bgp-evpn)# graceful-restart
```

8. Do any of the following:
   - Enter the **graceful-restart** command and use the **purge-time** parameter to overwrite the default purge time value.

     ```
     device(config-bgp-evpn)# graceful-restart purge-time 300
     ```

   - Enter the **graceful-restart** command and use the **restart-time** parameter to overwrite the default restart time advertised to graceful restart-capable neighbors.

     ```
     device(config-bgp-evpn)# graceful-restart restart-time 180
     ```

   - Enter the **graceful-restart** command and use the **stale-routes-time** parameter to overwrite the default amount of time that a helper device will wait for an EOR message from a peer.

     ```
     device(config-bgp-evpn)# graceful-restart stale-routes-time 100
     ```

The following example enables the graceful restart feature.

```
device# configure terminal
device(config)# rbridge-id 1
device(config-rbridge-id-1)# router bgp
device(config-bgp-router)# local-as 1
device(config-bgp-router)# neighbor 2001:db8:93e8:cc00::1 remote-as 2
device(config-bgp-router)# address-family l2vpn evpn
device(config-bgp-evpn)# graceful-restart
```

The following example enables the graceful restart feature and sets the purge time to 300 seconds, overwriting the default value.

```
device# configure terminal
device(config)# rbridge-id 1
device(config-rbridge-id-1)# router bgp
device(config-bgp-router)# local-as 1
device(config-bgp-router)# neighbor 2001:db8:93e8:cc00::1
device(config-bgp-router)# address-family l2vpn evpn
device(config-bgp-evpn)# graceful-restart purge-time 180
```

The following example enables the graceful restart feature and sets the restart time to 180 seconds, overwriting the default value.

```
device# configure terminal
device(config)# rbridge-id 1
device(config-rbridge-id-1)# router bgp
device(config-bgp-router)# local-as 1
device(config-bgp-router)# neighbor 2001:db8:93e8:cc00::1
device(config-bgp-router)# address-family l2vpn evpn
device(config-bgp-evpn)# graceful-restart restart-time 180
```

The following example enables the graceful restart feature and sets the stale-routes time to 100 seconds, overwriting the default value.

```
device# configure terminal
device(config)# rbridge-id 1
device(config-rbridge-id-1)# router bgp
device(config-bgp-router)# local-as 1
device(config-bgp-router)# neighbor 2001:db8:93e8:cc00::1
device(config-bgp-router)# address-family l2vpn evpn
device(config-bgp-evpn)# graceful-restart stale-routes-time 100
```

Configuring BGP peer groups for the L2VPN EVPN address family

A peer group can be created and activated in the L2VPN EVPN address family configuration mode.

1. Enter the **configure terminal** command to access global configuration mode.

   ```
   device# configure terminal
   ```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

```
device(config)# rbridge-id 1
```

3. Enter the **router bgp** command to enable BGP routing.

```
device(config-rbridge-id-1)# router bgp
```

4. Enter the **local-as** command to configure the autonomous system number (ASN) in which your device resides.

```
device(config-bgp-router)# local-as 1000
```

5. Enter the **neighbor** *peer-group-name* **peer-group** command to create a peer group.

```
device(config-bgp-router)# neighbor mypeergroup1 peer-group
```

6. Enter the **neighbor** *peer-group-name* **remote-as** command to specify the ASN of the peer group.

```
device(config-bgp-router)# neighbor mypeergroup1 remote-as 11
```

7. Enter the **neighbor** *ipv6-address* **peer-group** command to associate a neighbor with the peer group.

```
device(config-bgp-router)# neighbor 2001:2018:8192::125 peer-group mypeergroup1
```

8. Enter the **neighbor** *ipv6-address* **peer-group** command to associate a second neighbor with the peer group.

```
device(config-bgp-router)# neighbor 2001:2018:8192::124 peer-group mypeergroup1
```

9. Enter the **address-family l2vpn evpn** command to enter L2VPN EVPN address family configuration mode.

```
device(config-bgp-router)# address-family l2vpn evpn
```

10. Enter the **neighbor** *peer-group-name* **activate** command to establish a BGP EVPN session with the peer group.

```
device(config-bgp-evpn)# neighbor mypeergroup1 activate
```

The following example creates a peer group, specifying two neighbors to belong to the peer group, and activates the peer group in L2VPN EVPN.

```
device# configure terminal
device(config)# rbridge-id 1
device(config-rbridge-id-1)# router bgp
device(config-bgp-router)# local-as 1000
device(config-bgp-router)# neighbor mypeergroup1 peer-group
device(config-bgp-router)# neighbor mypeergroup1 remote-as 11
device(config-bgp-router)# neighbor 2001:2018:8192::125 peer-group mypeergroup1
device(config-bgp-router)# neighbor 2001:2018:8192::124 peer-group mypeergroup1
device(config-bgp-router)# address-family l2vpn evpn
device(config-bgp-evpn)# neighbor mypeergroup1 activate
```

Configuring a route reflector client for the L2VPN EVPN address family

A BGP peer can be configured as a route reflector (RR) client for the L2VPN EVPN address family. When iBGP is used for BGP EVPN in an IP Fabric, spine switches are configured as RRs and leaf switches are configured as RR clients. The following task configures an RR client.

1. Enter the **configure terminal** command to access global configuration mode.

```
device# configure terminal
```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

```
device(config)# rbridge-id 1
```

3. Enter the **router bgp** command to enable BGP routing.

```
device(config-rbridge-id-1)# router bgp
```

4. Enter the **local-as** command to configure the autonomous system number (ASN) in which your device resides.

```
device(config-bgp-router)# local-as 1000
```

5. Enter the **neighbor** *ipv4* **remote-as** command to specify the ASN in which the remote neighbor resides.

```
device(config-bgp-router)# neighbor 10.1.1.1 remote-as 3
```

6. Enter the **address-family l2vpn evpn** command to enter L2VPN EVPN address family configuration mode.

```
device(config-bgp-router)# address-family l2vpn evpn
```

7. Enter the **neighbor** *ip address* **route-reflector-client** command to configure a specified neighbor to be a route reflector client.

```
device(config-bgp-evpn)# neighbor 10.1.1.1 route-reflector-client
```

The following example configures a neighbor with the IP address 10.1.1.1 to be a route reflector client in L2VPN EVPN configuration mode.

```
device# configure terminal
device(config)# rbridge-id 1
device(config-rbridge-id-1)# router bgp
device(config-bgp-router)# local-as 1000
device(config-bgp-router)# neighbor 10.1.1.1 remote-as 3
device(config-bgp-router)# address-family l2vpn evpn
device(config-bgp-evpn)# neighbor 10.1.1.1 route-reflector-client
```

Disabling client-to-client reflection for the L2VPN EVPN address family

For iBGP, if the leaf switches are configured as route reflector clients and are connected in a full iBGP mesh, you can disable client-to-client reflection on the spine switch that is configured as a route reflector (RR).

1. Enter the **configure terminal** command to access global configuration mode.

```
device# configure terminal
```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

```
device(config)# rbridge-id 1
```

3. Enter the **router bgp** command to enable BGP routing.

```
device(config-rbridge-id-1)# router bgp
```

4. Enter the **local-as** command to configure the autonomous system number (ASN) in which your device resides.

```
device(config-bgp-router)# local-as 1000
```

5. Enter the **neighbor** *ipv4* **remote-as** command to specify the ASN in which the remote neighbor resides.

```
device(config-bgp-router)# neighbor 10.1.1.1 remote-as 3
```

6. Enter the **address-family l2vpn evpn** command to enter L2VPN EVPN address family configuration mode.

```
device(config-bgp-router)# address-family l2vpn evpn
```

7. Enter the **no client-to-client-reflection** command to disable client-to-client reflection.

```
device(config-bgp-evpn)# no client-to-client-reflection
```

The following example disables client-to-client reflection in L2VPN EVPN configuration mode.

```
device# configure terminal
device(config)# rbridge-id 1
device(config-rbridge-id-1)# router bgp
device(config-bgp-router)# local-as 1000
device(config-bgp-router)# neighbor 10.1.1.1 remote-as 3
device(config-bgp-router)# address-family l2vpn evpn
device(config-bgp-evpn)# no client-to-client-reflection
```

Disabling the BGP AS_PATH check function for the L2VPN EVPN address family

A device can be configured so that the AS_PATH check function for routes learned from a specific peer is disabled, and routes that contain the recipient BGP speaker's AS number are not rejected. For eBGP, with leaves in one AS and spines in another AS, the AS_PATH check function can be disabled for a leaf switch, so that route updates from the Spine layer, containing routes from the other leaves, are not discarded by the receiving leaf.

1. Enter the **configure terminal** command to access global configuration mode.

```
device# configure terminal
```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

```
device(config)# rbridge-id 1
```

3. Enter the **router bgp** command to enable BGP routing.

```
device(config-rbridge-id-1)# router bgp
```

4. Enter the **local-as** command to configure the autonomous system number (ASN) in which your device resides.

```
device(config-bgp-router)# local-as 1000
```

5. Enter the **neighbor** *ipv6-address* **remote-as** command to specify the ASN in which the remote neighbor resides.

```
device(config-bgp-router)# neighbor 2001:db8:e0ff:783a::4 remote-as 2
```

6. Enter the **address-family l2vpn evpn** command to enter L2VPN EVPN address family configuration mode.

```
device(config-bgp-router)# address-family l2vpn evpn
```

7. Enter the **neighbor** *ipv6 address* **activate** command to establish a BGP EVPN session with the eBGP neighbor.

```
device(config-bgp-evpn)# neighbor 2001:db8:e0ff:783a::4 activate
```

8. Enter the **neighbor** *ipv6-address* **allowas-in** command and specify a **number** to disable the BGP AS_PATH check function, and specify the number of times that the AS path of a received route may contain the recipient BGP speaker's AS number and still be accepted.

```
device(config-bgp-evpn)# neighbor 2001:db8:e0ff:783a::4 allowas-in 1
```

This example specifies that the AS path of a received route may contain the recipient BGP speaker's AS number once and still be accepted in L2VPN EVPN.

```
device# configure terminal
device(config)# rbridge-id 1
device(config-rbridge-id-1)# router bgp
device(config-bgp-router)# local-as 1000
device(config-bgp-router)# neighbor 2001:db8:e0ff:783a::4 remote-as 2
device(config-bgp-router)# address-family l2vpn evpn
device(config-bgp-evpn)# neighbor 2001:db8:e0ff:783a::4 activate
device(config-bgp-evpn)# neighbor 2001:db8:e0ff:783a::4 allowas-in 1
```

Enabling the BGP AS_PATH check function for the sender BGP speaker

A device can be configured so that a BGP sender speaker does not send routes with an AS path that contains the ASN of the receiving speaker.

NOTE
This task enforces the outbound BGP AS_PATH check function for the BGP IPv4 unicast. This feature is also supported for the BGP address-family IPv6 unicast and BGP address-family L2VPN EVPN address families.

1. Enter the **configure terminal** command to access global configuration mode.

   ```
   device# configure terminal
   ```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

   ```
   device(config)# rbridge-id 1
   ```

3. Enter the **router bgp** command to enable BGP routing.

   ```
   device(config-rbridge-id-1)# router bgp
   ```

4. Enter the **local-as** command to configure the autonomous system number (ASN) in which your device resides.

   ```
   device(config-bgp-router)# local-as 1000
   ```

5. Enter the **neighbor** *ip-address* **remote-as** command to specify the ASN in which the remote neighbor resides.

   ```
   device(config-bgp-router)# neighbor 10.1.1.1 remote-as 2
   ```

6. Enter the **address-family ipv4 unicast** command to enter BGP address-family IPv4 unicast configuration mode.

   ```
   device(config-bgp-router)# address-family ipv4 unicast
   ```

7. Enter the **neighbor** *ip address* **enable-peer-as-check** command to enforce the outbound AS_PATH check function for the IPv4 address family.

   ```
   device(config-bgp-ipv4u)# neighbor 10.1.1.1 enable-peer-as-check
   ```

The following example enables the outbound AS_PATH check function for the BGP IPv4 unicast address family for a particular peer.

```
device# configure terminal
device(config)# rbridge-id 1
device(config-rbridge-id-1)# router bgp
device(config-bgp-router)# local-as 1000
device(config-bgp-router)# neighbor 10.1.1.1 remote-as 2
device(config-bgp-router)# address-family ipv4 unicast
device(config-bgp-ipv4u)# neighbor 10.1.1.1 enable-peer-as-check
```

The following example enables the outbound AS_PATH check function for the BGP IPv6 unicast address family for a particular peer.

```
device# configure terminal
device(config)# rbridge-id 1
device(config-rbridge-id-1)# router bgp
device(config-bgp-router)# local-as 1000
device(config-bgp-router)# neighbor 2001:2018:8192::125 remote-as 2
device(config-bgp-router)# address-family ipv6 unicast
device(config-bgp-ipv6u)# neighbor 2001:2018:8192::125 enable-peer-as-check
```

The following example enables the outbound AS_PATH check function for the L2VPN EVPN address family for a particular peer.

```
device# configure terminal
device(config)# rbridge-id 1
device(config-rbridge-id-1)# router bgp
device(config-bgp-router)# local-as 1000
device(config-bgp-router)# neighbor 10.1.1.1 remote-as 2
device(config-bgp-router)# address-family l2vpn evpn
device(config-bgp-evpn)# neighbor 10.1.1.1 enable-peer-as-check
```

A neighbor reset is required once this task is complete.

Configuring Multi-VRF for BGP EVPN

Multi-VRF can be configured for BGP EVPN using the symmetric routing model. In this task, the device is enabled to advertise routes for Multi-VRF through a single BGP EVPN session.

1. Enter the **configure terminal** command to access global configuration mode.

   ```
   device# configure terminal
   ```

2. Enter the **interface vlan** command and specify a value to create a VLAN.

   ```
   device(config)# interface vlan 10
   ```

   Creates a VLAN with an ID of 10.

3. Enter the **exit** command to return to global configuration mode.

   ```
   device(config-Vlan-10)# exit
   ```

4. Enter the **interface vlan** command and specify a value to create a VLAN.

   ```
   device(config)# interface vlan 100
   ```

   Creates a VLAN with an ID of 100.

5. Enter the **exit** command to return to global configuration mode.

   ```
   device(config-Vlan-100)# exit
   ```

6. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

   ```
   device(config)# rbridge-id 1
   ```

7. Enter the **vrf** command and specify a name to create a VRF.

   ```
   device(config-rbridge-id-1)# vrf red
   ```

8. Enter the **rd** command and specify a value to create a route distinguisher for VRF red.

   ```
   device(config-vrf-red)# rd 101:1
   ```

9. Enter the **vni** command and specify a value to add a Layer 3 virtual network identifier (VNI) for VRF red.

```
device(config-vrf-red)# vni 100
```

   BGP includes the configured Layer 3 VNI and its associated MAC address in all EVPN updates.

10. Enter the **address-family unicast** command to enable IPv4 address-family configuration mode for VRF routing.

```
device(config-vrf-red)# address-family ipv4 unicast
```

11. Enter the **route-target** command with the **import** *value* and **evpn** keywords.

```
device(vrf-red-ipv4-unicast)# route-target import 32767:123 evpn
```

   Specifies the import route-target community attribute and assigns the local ASN number 32767:123 to the route. The EVPN route target is specified, causing BGP to advertise IPv4 VRF routes via the EVPN address family.

12. Enter the **route-target** command with the **export** *value* and **evpn** keywords.

```
device(vrf-red-ipv4-unicast)# route-target export 32767:123 evpn
```

   Specifies the export route-target community attribute and assigns the local ASN number 32767:123 to the route. The EVPN route target is specified, causing BGP to advertise IPv4 VRF routes via the EVPN address family.

13. Enter the **exit** command until you return to global configuration mode.

```
device(vrf-red-ipv4-unicast)# exit
```

14. Enter the **interface ve** command to configure a virtual Ethernet (VE) interface.

```
device(config)# interface ve 10
```

15. Enter the **vrf forwarding** command.

```
device(config-Ve-10)# vrf forwarding red
```

16. Enter the **ip address** command and specify an IP address.

```
device(config-Ve-10)# ip address 10.10.10.1/24
```

17. Enter the **no shutdown** command to enable the interface.

```
device(config-Ve-10)# no shutdown
```

18. Enter the **exit** command to return to global configuration mode.

```
device(config-Ve-10)# exit
```

19. Enter the **router bgp** command to enable BGP routing.

```
device(config-rbridge-id-1)# router bgp
```

20. Enter the **address-family ipv4 unicast** command with the **vrf** keyword, specifying a VRF name, to enable IPv4 unicast address-family mode for the VRF instance.

```
device(config-bgp-router)# address-family ipv4 unicast vrf red
```

The following example configures Multi-VRF for BGP EVPN. As shown below, the device is enabled to advertise routes for Multi-VRF through a single BGP EVPN session. BGP includes the configured Layer 3 VNI and its associated MAC address in all EVPN updates. The EVPN route target is specified, causing BGP to advertise IPv4 VRF routes via the EVPN address family.

```
device# configure terminal
device(config)# interface vlan 10
device(config-Vlan-10)# exit
device(config)# interface vlan 100
device(config-Vlan-100)# exit
device(config)# rbridge-id 1
device(config-rbridge-id-1)# vrf red
device(config-vrf-red)# rd 101:1
device(config-vrf-red)# vni 100
device(config-vrf-red)# address-family ipv4 unicast
device(vrf-red-ipv4-unicast)# route-target import 32767:123 evpn
device(vrf-red-ipv4-unicast)# route-target export 32767:123 evpn
device(vrf-red-ipv4-unicast)# exit
device(config)# interface ve 10
device(config-Ve-10)# vrf forwarding red
device(config-Ve-10)# ip address 10.10.10.1/24
device(config-Ve-10)# no shutdown
device(config-Ve-10)# exit
device(config-rbridge-id-1)# router bgp
device(config-bgp-router)# address-family ipv4 unicast vrf red
```

Configuration commands supporting BGP EVPN

The following configuration commands support BGP EVPN.

For more details, refer to the *Command Reference*.

RBridge ID configuration mode

- evpn-instance

BGP configuration mode

- address-family l2vpn evpn

BGP address-family L2VPN EVPN configuration mode

- client-to-client-reflection
- graceful-restart (BGP)
- neighbor activate
- neighbor allowas-in
- neighbor enable-peer-as-check
- neighbor maximum-prefix
- neighbor next-hop-unchanged
- neighbor route-map
- neighbor route-reflector-client
- neighbor send-community
- retain route-target all

- vtep-discovery

EVPN instance configuration mode

- duplicate-mac-timer
- evpn
- rd auto (EVPN)
- route-target (EVPN)
-
- vni (EVPN)
- vni add
- vni remove

VNI configuration mode

- rd (VNI)
- route-target (VNI)

VRF configuration mode

- rd (VRF)
- route-target (VRF)
-
- vni (VRF)

Port-channel configuration mode

- esi

VXLAN overlay-gateway site configuration mode

- mac-learning protocol bgp

Show commands supporting BGP EVPN

The following show commands support BGP EVPN.

For more details, refer to the *Command Reference*.

- show bgp evpn dampened-routes
- show bgp evpn interface port-channel
- show bgp evpn interface tunnel
- show bgp evpn l2route detail
- show bgp evpn l2route next-hop

- show bgp evpn l2route summary
- show bgp evpn l2route type arp
- show bgp evpn l2route type auto-discovery
- show bgp evpn l2route type ethernet-segment
- show bgp evpn l2route type inclusive-multicast
- show bgp evpn l2route type mac
- show bgp evpn l2route type nd
- show bgp evpn l2route unreachable
- show bgp evpn l3vni
- show bgp evpn neighbors
- show bgp evpn neighbors advertised-routes detail
- show bgp evpn neighbors advertised-routes type
- show bgp evpn neighbors routes best
- show bgp evpn neighbors routes detail
- show bgp evpn neighbors routes not-installed-best
- show bgp evpn neighbors routes type
- show bgp evpn neighbors routes unreachable
- show bgp evpn neighbors routes-summary
- show bgp evpn routes
- show bgp evpn routes best
- show bgp evpn routes detail
- show bgp evpn routes local
- show bgp evpn routes next-hop
- show bgp evpn routes no-best
- show bgp evpn routes not-installed-best
- show bgp evpn routes rd
- show bgp evpn routes rd type
- show bgp evpn routes summary
- show bgp evpn routes type arp
- show bgp evpn routes type auto-discovery
- show bgp evpn routes type ethernet-segment
- show bgp evpn routes type inclusive-multicast
- show bgp evpn routes type ipv4-prefix
- show bgp evpn routes type ipv6-prefix
- show bgp evpn routes type mac
- show bgp evpn routes type nd
- show bgp evpn routes unreachable
- show bgp evpn summary
- show mac-address-table

- **show mac-address-table count evpn**
- **show mac-address-table evpn**

BFD

Bidirectional Forwarding Detection overview

Bidirectional Forwarding Detection (BFD) is a unified detection mechanism used to rapidly detect link faults. BFD improves network performance by providing fast forwarding path failure detection times.

BFD provides rapid detection of the failure of a forwarding path by checking that the next-hop device is alive. When BFD is not enabled, it can take time to detect that a neighboring device is not operational. This causes packet loss due to incorrect routing information at a level unacceptable for real-time applications such as VOIP and video over IP.

Using BFD, you can detect a forwarding path failure in 150 milliseconds.

A BFD session is automatically established when a neighbor is discovered for a protocol, provided that BFD is enabled on the interface on which the neighbor is detected and BFD is also enabled for the protocol at the interface level or globally. Once a session is established, each device transmits control messages at a high rate of speed that is negotiated by the devices during the session setup. To provide a detection time of 150 milliseconds, it is necessary to process 20 messages per second of about 70 to 100 bytes each per session. A similar number of messages also need to be transmitted out per session. Once a session is established, that same message is continuously transmitted at the negotiated rate and a check is made that the expected control message is received at the agreed frequency from the neighbor. If the agreed upon messages are not received from the neighbor within a negotiated timeout period, the neighbor is considered to be down.

BFD can provide failure detection on any kind of path between systems, including direct physical links, multihop routed paths, and tunnels. Multiple BFD sessions can be established between the same pair of systems when multiple paths between them are present in at least one direction, even if a lesser number of paths are available in the other direction.

NOTE

BFD session establishment on an interface does not start until 5 seconds after the interface comes up. The reason for this delay is to ensure that the link is not affected by unstable link conditions which could cause BFD to flap. This delay time is not user configurable.

For a single-hop session, the BFD Control Message is a UDP message with destination port 3784. For a multihop session, the BFD Control Message is a UDP message with destination port 4784 sent over IPv4 or IPv6, depending on which data forwarding path failure BFD is trying to detect.

NOTE
The source port for BFD control packets is in the range 49152 through 65535. The source port number is unique among all BFD sessions on the system.

NOTE
For single-hop sessions, all BFD control packets are sent with a time to live (TTL) or hop limit value of 255. All received BFD control packets are discarded if the received TTL or hop limit is not equal to 255.

General BFD considerations and limitations

There are a number of general points to consider when configuring BFD:

- BFD is not supported on Layer 3 port channels.
- BFD is supported on Layer 2 port channels used by SVI sessions. Refer to the *Extreme Network OS Layer 2 Switching Configuration Guide* for more information on Layer 2 port channels. For trunk ports, active BFD sessions are started on the line card (LC) with the primary port. Inactive BFD sessions are started on all other LCs with secondary member ports. The BFD state machine is replicated on all LCs. Inactive sessions on other LCs receive control packets that arrive from alternate paths. Session timeouts are triggered if control packets are not received on any path.
- BFD protocol version 1 is supported. BFD version 0 is not supported. BFD version 1 and BFD version 0 are not compatible.
- BFD single-hop sessions always use the primary IP address as the source address. Secondary IP addresses cannot be used as source addresses on single-hop sessions.

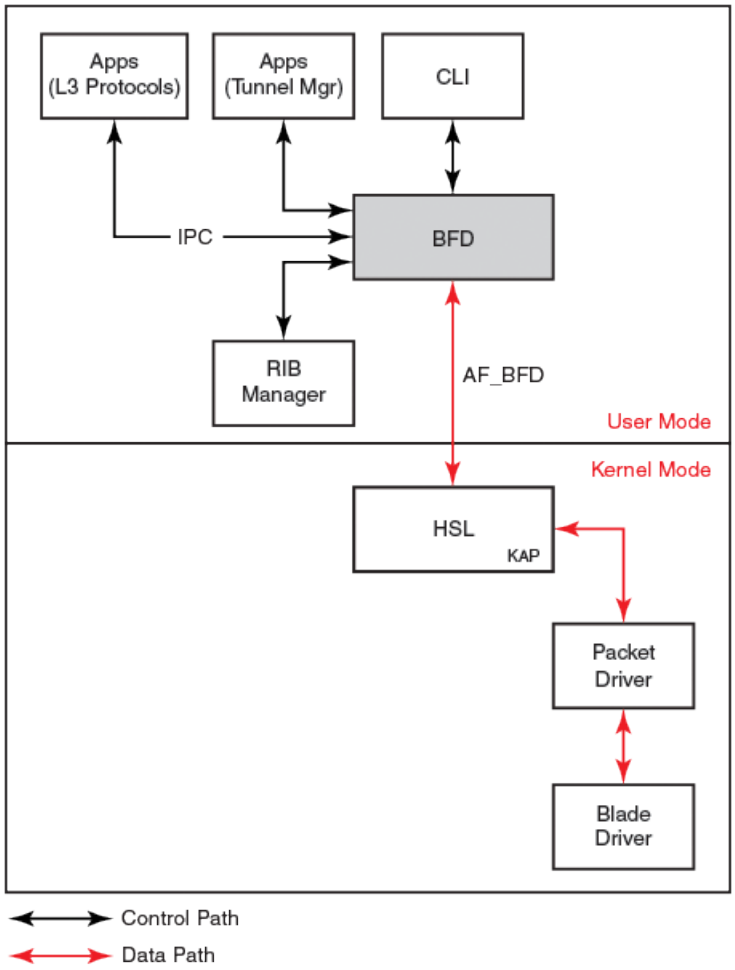
Refer to BFD considerations and limitations for Layer 3 protocols on page 323 and the *BFD considerations and limitations for static routes* section of the "IP Route Policy" chapter for more information on BFD considerations and limitations.

BFD on Network OS hardware platforms

A single instance of BFD runs on stackable switch platforms, and BFD sessions cannot be offloaded. On chassis-based platforms support for offloading BFD sessions to line cards is provided, helping to achieve scalability.

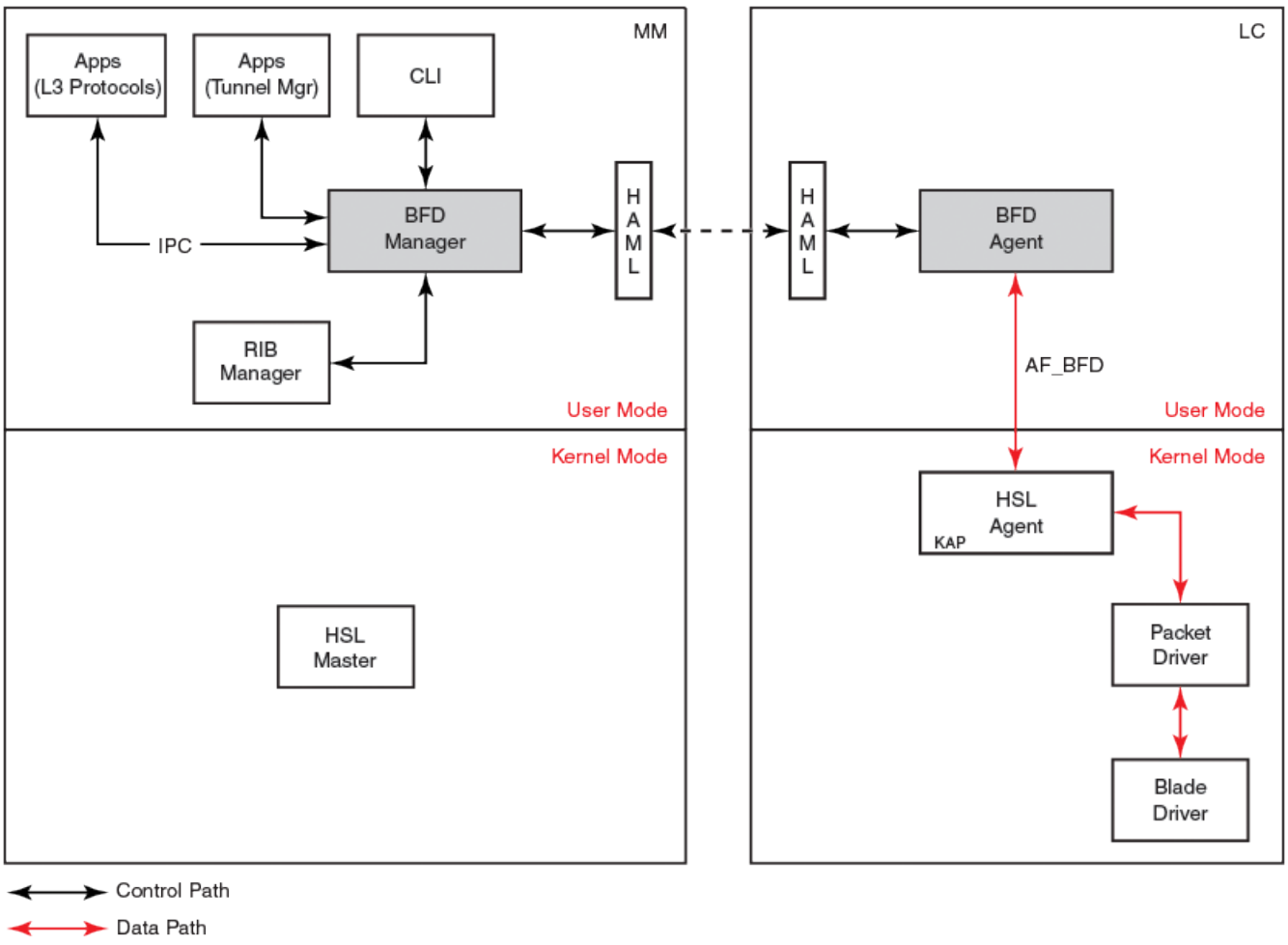
The figure below shows BFD running on a stackable switch platform, where BFD sessions are not offloaded. BFD is responsible for providing the forwarding path failure detection service to a routing protocol.

FIGURE 34 BFD module on a stackable switch



The figure below shows BFD running on a chassis-based platform, where BFD sessions are offloaded to a LC. BFD is responsible for actually detecting the failure of the forwarding path.

FIGURE 35 BFD module on a chassis-based platform



BFD for Layer 3 protocols

BFD can be used by Layer 3 protocols for rapid failure detection in the forwarding path between two adjacent routers, including the interfaces, data links, and forwarding planes.

BFD can be configured for use with the following protocols:

- OSPFv2
- OSPFv3
- BGP4
- BGP4+

BFD must be enabled at both the interface and routing protocol levels. BFD asynchronous mode, which depends on the sending of BFD control packets between two systems to activate and maintain BFD neighbor sessions between routers, is supported. Therefore, in order for a BFD session to be created, BFD must be configured on both BFD peers.

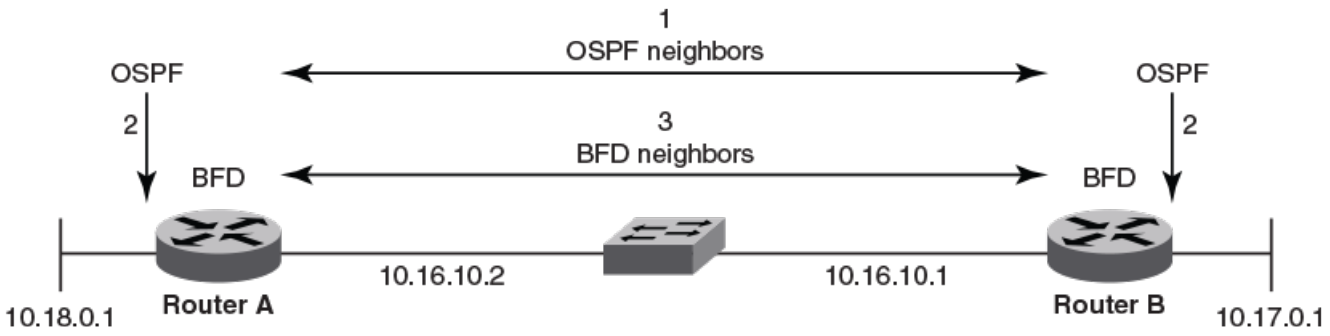
Once BFD is enabled on the interfaces and at the router level for the appropriate routing protocols, a BFD session is created. BFD timers are then negotiated, and the BFD peers begin to send BFD control packets to each other at the negotiated interval.

BFD provides a single point of forwarding path monitoring. This means that when more than one Layer 3 application wants to monitor a single host, BFD runs a single session for that host and provides the status to multiple applications, instead of multiple applications running individual sessions to the host.

By sending rapid failure detection notices to the routing protocols in the local device to initiate the routing table recalculation process, BFD contributes to greatly reducing overall network convergence time.

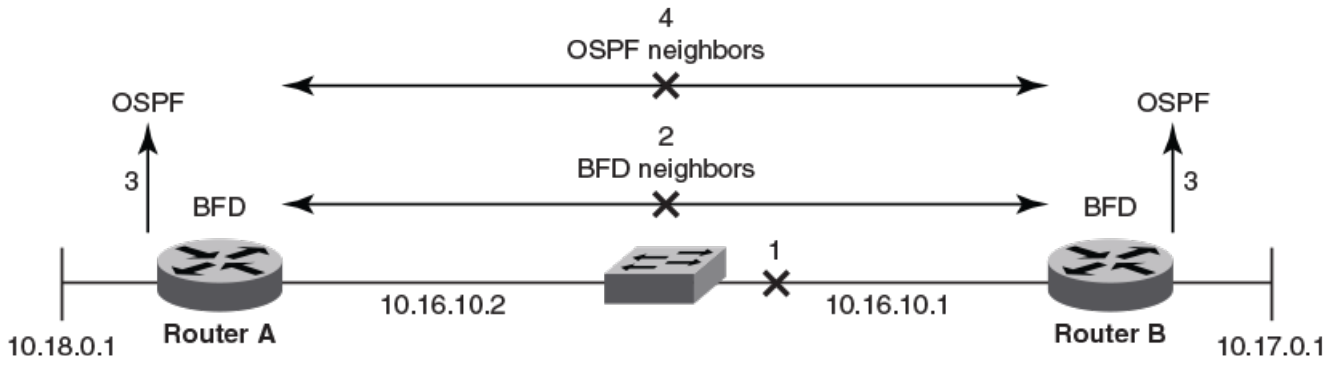
The following figure shows the establishment of a BFD session where OSPF discovers a neighbor and sends a request to BFD requesting that a BFD neighbor session be created with the OSPF neighbor router.

FIGURE 36 Establishing a BFD neighbor session



The following figure shows the termination of a BFD neighbor session after a failure occurs in the network.

FIGURE 37 Termination of a BFD neighbor session



BFD considerations and limitations for Layer 3 protocols

There are a number of things to consider when configuring BFD for Layer 3 protocols:

- BFD is not supported on Layer 3 port channels. For information on BFD support on Layer 2 port channels, refer to the *Extreme Network OS Layer 2 Switching Configuration Guide*.
- BFD supports both single-hop and multihop sessions.

- For single-hop sessions, the IP address that matches the subnet of the destination IP address is always used as the source IP address of the BFD control packet. If the source IP address is changed, the session is brought down unless another corresponding address with the same subnet is available.
- For single-hop sessions, an IPv6 address that matches the prefix and scope of the destination IPv6 address is used as the source IPv6 address of the BFD control packet.
- For multihop sessions, BFD clients provide the source IP address and BGP is notified of any change to the source IP address of the BFD control packet.
- Multicast or anycast address IP addresses can not be used as source IP addresses.
- BFD establishes only a single BFD session per data protocol path (IPv4 or IPv6) regardless of the number of protocols that want to utilize it.
- Registration is global across all VRFs, even if a neighbor does not support BFD.
- Inactive sessions are created when BFD is not enabled on a remote device that contains the destination IP address. Sessions created are in the Admin Down state and are transmitted at a slower rate than configured values.
- BFD sessions can be established on both primary and secondary IP and IPv6 addresses.
- BFD sessions can be established on link-local IPv6 addresses.
- Changing the BFD parameters does not reset the current BFD session.
- When BFD notifies BGP or OSPF that a session has transitioned from up to down, the protocol does not immediately bring down the session if the holdover timer is configured. The protocol waits until the period of time specified for the holdover timer has expired. If BFD declares a session up before this period of time expires, no action is taken by the protocol.
- If you unconfigure a BFD session that is in the up state, OSPF or BGP tell BFD to delete the session and set the reason as Admin Down. Upon receipt of this notification, BFD deletes the session and communicates this change to the remote BFD peer. The remote BFD neighbor keeps the session in the down state and propagates Remote Admin Down event to the routing protocols.

BFD for Layer 3 protocols on virtual Ethernet interfaces

BFD can be configured for Layer 3 protocols on virtual Ethernet (VE) interfaces.

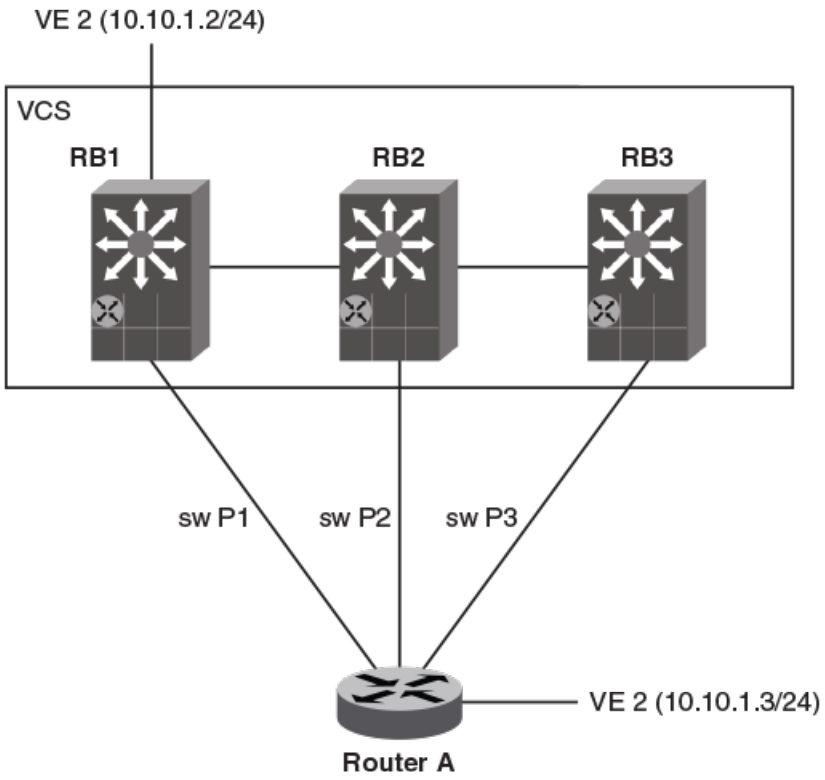
When configuring BFD for Layer 3 protocols on VE interfaces, one of the physical ports is chosen to set up the session. The physical port used for setting up the BFD session is allocated after Address Resolution Protocol (ARP) is resolved for that neighbor.

If a member of a VE port goes down and a new egress port is identified, the session is moved to another physical port. If a new egress port is not identified, BFD tries to locate a destination IPv4 or IPv6 address. If the new egress port is not learned within the BFD detection time, the session is declared as down.

BFD support for Layer 3 protocols on VEs over VCS

When a BFD session is created, the RBridge where the session is created is designated as the “master RBridge”. This master RBridge runs the full BFD state machine. In the figure below, RB1 is the master RBridge and runs the BFD session. The RBridges are connected through an Inter-Switch Link (ISL) . If the ARP-resolved egress port is in RB2, RB1 begins hardware-assisted BFD packet transmission over the ISL connected to RB2. If the switch port, swP2, goes down or RB2 gets disconnected from the VCS, RB1 begins BFD packet transmission over the next ARP resolved egress port.

FIGURE 38 BFD on a virtual Ethernet interface over VCS



BFD packets received in switch ports of Remote RBridges are redirected to the master RBridge.

BFD for Layer 3 protocols on vLAGs

BFD can be configured for Layer 3 protocols on Virtual Link Aggregation Groups (vLAGs).

When configuring BFD on a vLAG, the master RBridge begins BFD packet transmission over the local member port. If the local member port goes down, the vLAG parent virtual Ethernet (VE) interface is reachable using the port channel member port of remote RBridges. This parent VE interface remains in the Up state throughout.

If the master RBridge does not have any local member port, it begins hardware-assisted BFD packet transmission over an Inter-Switch Link (ISL) connected to a remote RBridge. If the vLAG secondary member port goes down, or the second RBridge is disconnected from the VCS, the master RBridge begins BFD packet transmission over another member port.

Configuring BFD on an interface

BFD can be configured on device interfaces. Repeat the steps in this procedure for each interface on which you want to configure BFD sessions.

NOTE
This task does not result in BFD session creation, but configures BFD session parameters for the interface.

1. Enter the **configure terminal** command to access global configuration mode.

```
device# configure terminal
```

2. Enter the **interface** command and specify an interface.

```
device(config)# interface tengigabitethernet 1/0/1
```

3. Enter the **bfd interval** command with the **min-rx** and **multiplier** keywords to configure BFD session parameters on the interface.

```
device(config-if-te-1/0/1)# bfd interval 110 min-rx 120 multiplier 15
```

The following example configures BFD on a specific 10-gigabit Ethernet interface by setting the baseline BFD session parameters on that interface.

```
device# configure terminal
device(config)# interface tengigabitethernet 1/0/1
device(config-if-te-1/0/1)# bfd interval 110 min-rx 120 multiplier 15
```

Disabling BFD on an interface

BFD can be disabled on device interfaces. Repeat the steps in this procedure for each interface onThe following figure which you want to disable BFD sessions.

1. Enter the **configure terminal** command to access global configuration mode.

```
device# configure terminal
```

2. Enter the **interface** command and specify an interface.

```
device(config)# interface tengigabitethernet 1/0/1
```

3. Enter the **bfd shutdown** command to disable BFD on the interface.

```
device(config-if-te-1/0/1)# bfd shutdown
```

The following example disables BFD on a specific 10-gigabit Ethernet interface.

```
device# configure terminal
device(config)# interface tengigabitethernet 1/0/1
device(config-if-te-1/0/1)# bfd shutdown
```

BFD for BGP

BFD support for BGP4 and BGP4+ can be configured so that BGP is a registered protocol with BFD and receives forwarding path detection failure messages from BFD.

BFD is supported for BGP and is disabled by default. When BFD for BGP is enabled, BFD rapidly detects faults on links between BGP peers and reports faults to BGP. BFD for BGP is supported for both for single-hop and multihop iBGP and eBGP sessions with either IPv4 or IPv6 neighbors in the default VRF and nondefault VRF instances. BFD behavior is identical for iBGP and eBGP single-hop and multihop sessions, and for IPv4 and IPv6 neighbors.

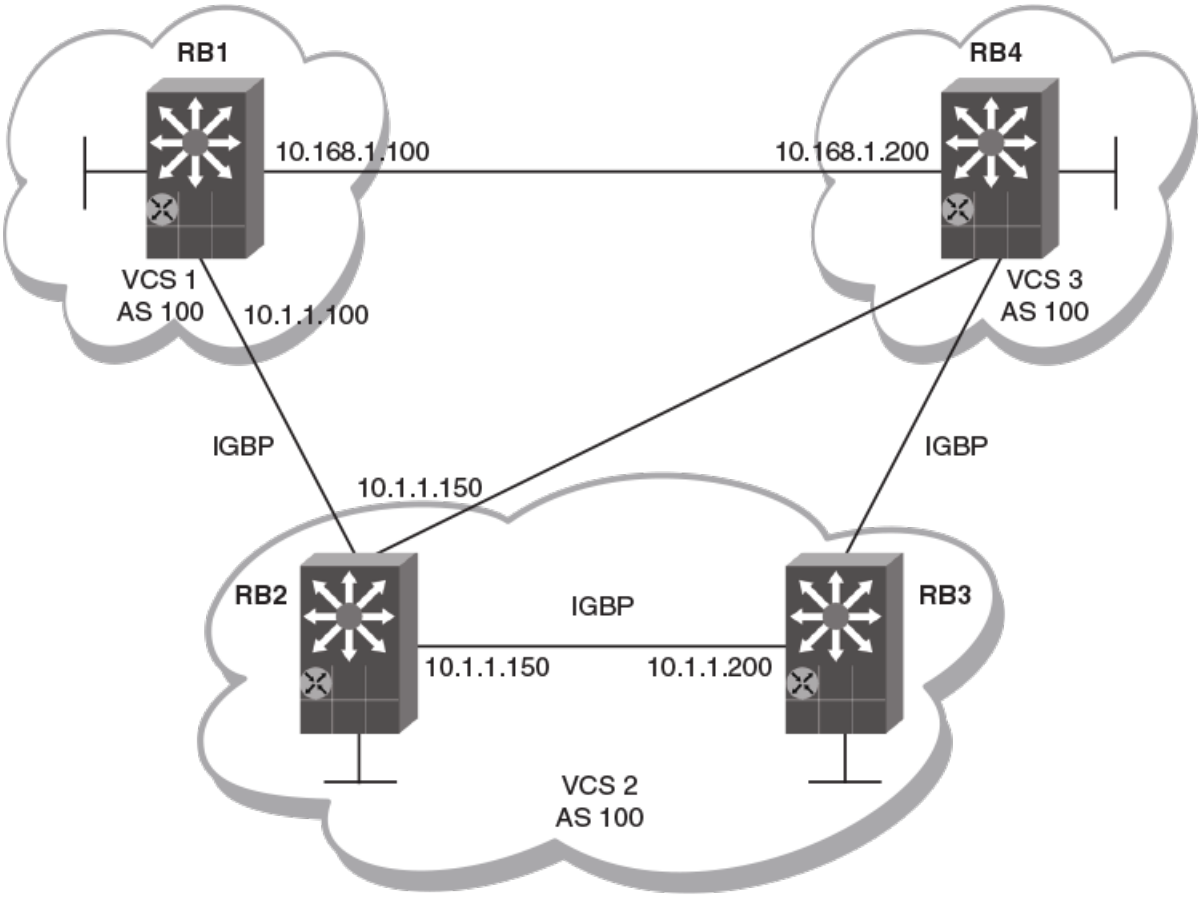
> NOTE
> BFD for unnumbered EMCP interfaces is supported for BGP in IP Fabric deployments. For further information, refer to the *Extreme Network OS IP Fabrics Configuration Guide*.

Consider the following when configuring BFD for BGP:

- Registration is global across all VRFs once BGP sends a registration message to BFD.
- BFD sessions for remote BGP neighbors are not triggered if BFD is not configured on these neighbors. Each neighbor can have its own transmit interval, receive internal, and detect multiplier. If the value is not configured, the configured global value is inherited.
- As soon as a BGP session enters the Established state, BGP requests that BFD start a BFD session.
- BFD sessions are maintained across a BGP graceful restart.

The figure below shows an IBGP session. This session running between RB1 and RB2 is a singlehop BFD session that tracks the remote address. The session running between RB1 and RB3 is a BFD multihop session that tracks the end points.

FIGURE 39 BFD for BGP



BFD for BGP session creation and deletion

When BGP requests that BFD start a BFD session, each session has associated BFD session parameters configured. Session parameter values are selected according to a specific hierarchy.

For multihop BGP neighbors, session parameter values are selected according to the following hierarchy:

- BFD session values configured at neighbor level
- BFD session values configured at BGP neighbor group level
- BFD session values configured at global BGP level
- Default values

For single-hop neighbors, interface-level BFD parameters are applied.

BFD sessions are deleted for a particular source or destination IPv4 or IPv6 address when a BGP session moves from the Established state to another BGP state.

BFD sessions are deleted when a BGP session for a neighbor is unconfigured locally. If you unconfigure a BGP session that is in an Established state and running a BFD session with a neighbor that is up, BGP sends a Cease message to the peer and deletes the BFD session. Upon receipt of this Cease message, the neighbor deletes the BFD session and moves to an Idle state.

Configuring BFD session parameters for BGP

BFD session parameters can be set globally for BGP-enabled interfaces.

1. Enter the **configure terminal** command to access global configuration mode.

```
device# configure terminal
```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

```
device(config)# rbridge-id 122
```

3. Enter the **router bgp** command to enable BGP routing.

```
device(config-rbridge-id-122)# router bgp
```

4. Enter the **bfd interval** command with the **min-rx** and **multiplier** keywords to configure BFD session parameters globally for BGP-enabled interfaces.

```
device(config-bgp-router)# bfd interval 110 min-rx 120 multiplier 15
```

NOTE
The **bfd interval** command is used for single-hop sessions only. Multihop sessions in BGP use either the values configured at the interface level using the **bfd interval** command or the default interval values.

5. Enter the **bfd holdover-interval** command and specify a value to set the BFD holdover interval globally for BGP-enabled interfaces.

```
device(config-bgp-router)# bfd holdover-interval 15
```

The following example configures BFD session parameters globally for BGP-enabled interfaces.

```
device# configure terminal
device(config)# rbridge-id 122
device(config-rbridge-id-122)# router bgp
device(config-bgp-router)# bfd interval 110 min-rx 120 multiplier 15
device(config-bgp-router)# bfd holdover-interval 15
```

Enabling BFD sessions for a specified BGP neighbor

BFD sessions can be configured for specified BGP neighbors.

1. Enter the **configure terminal** command to access global configuration mode.

```
device# configure terminal
```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

```
device(config)# rbridge-id 122
```

3. Enter the **router bgp** command to enable BGP routing.

```
device(config-rbridge-id-122)# router bgp
```

```
device(config)# router bgp
```

4. Enter the **neighbor bfd** command, specifying an IP address, to enable BFD sessions for the specified neighbor.

```
device(config-bgp-router)# neighbor 10.10.1.1 bfd
```

5. Enter the **neighbor bfd** command, specifying an IP address, with the **interval**, **min-rx**, and **multiplier** keywords to set the BFD session timer values for the specified BGP neighbor.

```
device(config-bgp-router)# neighbor 10.10.1.1 bfd interval 120 min-rx 140 multiplier 10
```

6. Enter the **neighbor bfd holdover-interval** command, specifying an IP address, and enter a value to set the BFD holdover interval for the specified BGP neighbor.

```
device(config-bgp-router)# neighbor 10.10.1.1 bfd holdover-interval 12
```

The following example enables a BFD session for a BGP neighbor with the IP address 10.10.1.1 and configures the BFD session parameters.

```
device# configure terminal
device(config)# rbridge-id 122
device(config-rbridge-id-122)# router bgp
device(config-bgp-router)# neighbor 10.10.1.1 bfd
device(config-bgp-router)# neighbor 10.10.1.1 bfd interval 120 min-rx 140 multiplier 10
device(config-bgp-router)# neighbor 10.10.1.1 bfd holdover-interval 12
```

Enabling BFD sessions for a specified BGP neighbor in a nondefault VRF

BFD sessions can be configured for specified BGP neighbors in a nondefault VRF instance.

1. Enter the **configure terminal** command to access global configuration mode.

```
device# configure terminal
```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

```
device(config)# rbridge-id 122
```

3. Enter the **router bgp** command to enable BGP routing.

```
device(config-rbridge-id-122)# router bgp
```

4. Enter the **address-family ipv4 unicast** command with the **vrf** keyword, specifying a VRF name, to enter BGP address-family IPv4 unicast VRF configuration mode.

```
device(config-bgp-router)# address-family ipv4 unicast vrf green
```

5. Enter the **neighbor bfd** command, specifying an IP address, to enable BFD sessions for the specified neighbor in a nondefault VRF instance.

```
device(config-bgp-ipv4u-vrf)# neighbor 10.10.1.1 bfd
```

6. Enter the **neighbor bfd** command, specifying an IP address, with the **interval**, **min-rx**, and **multiplier** keywords to set the BFD session timer values for the specified BGP neighbor in a nondefault VRF instance.

```
device(config-bgp-ipv4u-vrf)# neighbor 10.10.1.1 bfd interval 120 min-rx 140 multiplier 10
```

7. Enter the **neighbor bfd holdover-interval** command, specifying an IP address, and enter a value to set the BFD holdover interval for the specified BGP neighbor in a nondefault VRF instance.

```
device(config-bgp-ipv4u-vrf)# neighbor 10.10.1.1 bfd holdover-interval 12
```

The following example enables a BFD session for a BGP neighbor with the IP address 10.10.1.1 and configures the BFD session parameters in VRF instance “green”.

```
device# configure terminal
device(config)# rbridge-id 122
device(config-rbridge-id-122)# router bgp
device(config-bgp-router)# address-family ipv4 unicast vrf green
device(config-bgp-ipv4u-vrf)# neighbor 10.10.1.1 bfd
device(config-bgp-ipv4u-vrf)# neighbor 10.10.1.1 bfd interval 120 min-rx 140 multiplier 10
device(config-bgp-ipv4u-vrf)# neighbor 10.10.1.1 bfd holdover-interval 12
```

Enabling BFD sessions for a specified BGP peer group

BFD sessions can be configured for specified BGP peer groups.

1. Enter the **configure terminal** command to access global configuration mode.

   ```
   device# configure terminal
   ```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

   ```
   device(config)# rbridge-id 122
   ```

3. Enter the **router bgp** command to enable BGP routing.

   ```
   device(config-rbridge-id-122)# router bgp
   ```

4. Enter the **neighbor** *peer-group-name* **peer-group** command to create a peer group.

   ```
   device(config-bgp-router)# neighbor pg1 peer-group
   ```

5. Enter the **neighbor bfd** command, specifying a peer group, to enable BFD sessions for the specified peer group.

   ```
   device(config-bgp-router)# neighbor pg1 bfd
   ```

6. Enter the **neighbor bfd** command, specifying a peer group, with the **interval**, **min-rx**, and **multiplier** keywords to set the BFD session timer values for the specified BGP peer group.

   ```
   device(config-bgp-router)# neighbor pg1 bfd interval 200 min-rx 220 multiplier 25
   ```

7. Enter the **neighbor bfd holdover-interval** command, specifying a peer group, and enter a value to set the BFD holdover interval for the specified BGP peer group.

   ```
   device(config-bgp-router)# neighbor pg1 bfd holdover-interval 17
   ```

The following example enables a BFD session for a BGP peer group called “pg1” and configures the BFD session parameters.

```
device# configure terminal
device(config)# rbridge-id 122
device(config-rbridge-id-122)# router bgp
device(config-bgp-router)# neighbor pg1 peer-group
device(config-bgp-router)# neighbor pg1 bfd
device(config-bgp-router)# neighbor pg1 bfd interval 200 min-rx 220 multiplier 25
device(config-bgp-router)# neighbor pg1 bfd holdover-interval 17
```

Enabling BFD sessions for a specified BGP peer group in a nondefault VRF

BFD sessions can be configured for specified BGP peer groups in a nondefault VRF instance.

1. Enter the **configure terminal** command to access global configuration mode.

```
device# configure terminal
```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

```
device(config)# rbridge-id 122
```

3. Enter the **router bgp** command to enable BGP routing.

```
device(config-rbridge-id-122)# router bgp
```

4. Enter the **address-family ipv6 unicast** command with the **vrf** keyword, specifying a VRF name, to enter BGP address-family IPv6 unicast VRF configuration mode.

```
device(config-bgp-router)# address-family ipv6 unicast vrf red
```

5. Enter the **neighbor** *peer-group-name* **peer-group** command to create a peer group.

```
device(config-bgp-ipv6u-vrf)# neighbor pg1 peer-group
```

6. Enter the **neighbor bfd** command, specifying a peer group, to enable BFD sessions for the specified peer group in a nondefault VRF instance.

```
device(config-bgp-ipv6u-vrf)# neighbor pg1 bfd
```

7. Enter the **neighbor bfd** command, specifying a peer group name, with the **interval**, **min-rx**, and **multiplier** keywords to set the BFD session timer values for the specified BGP neighbor in a nondefault VRF instance.

```
device(config-bgp-ipv6u-vrf)# neighbor pg1 bfd interval 145 min-rx 155 multiplier 15
```

8. Enter the **neighbor bfd holdover-interval** command, specifying a peer group name, and enter a value to set the BFD holdover interval for the specified BGP peer group in a nondefault VRF instance.

```
device(config-bgp-ipv6u-vrf)# neighbor pg1 bfd holdover-interval 18
```

The following example enables a BFD session for a BGP peer group called “pg1”and configures the BFD session parameters for VRF instance “red”.

```
device# configure terminal
device(config)# rbridge-id 122
device(config-rbridge-id-122)# router bgp
device(config-bgp-router)# address-family ipv6 unicast vrf red
device(config-bgp-ipv6u-vrf)# neighbor pg1 peer-group
device(config-bgp-ipv6u-vrf)# neighbor pg1 bfd
device(config-bgp-ipv6u-vrf)# neighbor pg1 bfd interval 145 min-rx 155 multiplier 15
device(config-bgp-ipv6u-vrf)# neighbor pg1 bfd holdover-interval 18
```

BFD for OSPF

BFD support for OSPFv2 and OSPFv3 can be configured so that OSPF is a registered protocol with BFD and receives forwarding path detection failure messages from BFD.

BFD sessions can rapidly detect link faults and notify OSPF so that it quickly responds to network topology changes. BFD is supported for OSPF and is disabled by default.

Consider the following when configuring BFD for OSPF:

- OSPF uses single-hop BFD sessions.
- Virtual links are not supported.
- BFD for OSPF can be enabled in interface subtype configuration mode, OSPF VRF configuration mode, or OSPFv3 configuration mode. BFD must be enabled at both the interface level and global level to enable BFD for OSPF sessions.
- OSPF sends BFD a registration message even if BFD is not enabled on an interface or globally at the router OSPF level.
- Registration is global across all VRFs.
- BFD sessions are maintained across OSPF graceful restart.
- BFD for OSPF does not support authentication for BFD.

BFD for OSPF session creation and deletion

OSPF neighbors are discovered dynamically using the OSPF hello protocol. However, in the case of non-broadcast multiple access (NBMA) and point to multipoint (P2MP), neighbors must be manually configured. Regardless of whether neighbors are discovered dynamically or statically configured, all neighbors are associated with an interface. When BFD is enabled on an interface and at the global level, BFD sessions are created for all OSPF neighbors that are in greater than the 2-way state. A BFD session is created once it progresses to the INIT state.

Each interface can have its own BFD timers. OSPF does not influence the BFD timer value for the session. The configured BFD value, or the default value, is used by the BFD module when creating the session. A single-hop session is created for OSPF neighbors.

NOTE
When BFD for an OSPF session is configured, the normal OSPF hello mechanism is not disabled.

NOTE
OSPF neighbor sessions do not flap when BFD is enabled or disabled on the interface where the OSPF session is associated.

NOTE
OSPF assumes full connectivity between all systems on multi-access media such as LANs. If BFD is running on only a subset of systems on such a network, the assumptions of the control protocol may be violated, with unpredictable results.

OSPF BFD session deletion can happen in the following instances:

- If an OSPF neighbor session moves to a state below 2-way, OSPF triggers a BFD session deletion setting the reason as Path Down. When BFD receives this notification, it deletes the corresponding session. The remote BFD neighbor detects this as a detection timer expiry and propagates this session down to OSPF to terminate the session.
- If OSPF is disabled on an interface, all BFD sessions associated with each neighbor are deleted.
- If BFD is administratively disabled on an interface, BFD reports this event as a port change notification to OSPF. Upon receipt of this notification, a BFD session deletion for each neighbor is sent and BFD removes these sessions if no other client is using

the same sessions. BFD communicates this change to the remote peer. When the remote BFD peer receives this Remote Admin Down event, it propagates this event to the OSPF protocol. OSPF does not provide any action for this event.

Enabling BFD on a specified OSPFv2-enabled interface

BFD sessions can be configured on one or more OSPFv2-enabled interfaces.

BFD sessions are initiated on specified OSPFv2-enabled interfaces only if BFD is enabled globally using the **bfd** command in OSPF VRF configuration mode.

1. Enter the **configure terminal** command to access global configuration mode.

```
device# configure terminal
```

2. Enter the **interface** command and specify an interface.

```
device(config)# interface tengigabitethernet 101/0/10
```

3. Enter the **ip ospf bfd** command to enable BFD on the specified interface.

```
device(config-if-te-101/0/10)# ip ospf bfd
```

The following example enables BFD on a specified OSPFv2-enabled 10-gigabit Ethernet interface.

```
device# configure terminal
device(config)# tengigabitethernet 101/0/10
device(config-if-te-101/0/10)# ip ospf bfd
```

Configuring BFD for OSPFv2 globally

BFD for OSPFv2 can be configured globally on interfaces where BFD has been configured.

BFD must be configured using the **ip ospf bfd** command on each OSPFv2 interface to initiate BFD sessions.

1. Enter the **configure terminal** command to access global configuration mode.

```
device# configure terminal
```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

```
device(config)# rbridge-id 122
```

3. Enter the **router ospf** command to enter OSPF router configuration mode and enable OSPFv2 on the device.

```
device(config-rbridge-id-122)# router ospf
```

4. Enter the **bfd** command to enable BFD globally.

```
device(config-router-ospf-vrf-default-vrf)# bfd
```

5. Enter the **bfd holdover-interval** command to set the BFD holdover interval.

```
device(config-router-ospf-vrf-default-vrf)# bfd holdover-interval 12
```

The following example enables BFD globally and sets the BFD holdover interval to 12.

```
device# configure terminal
device(config)# rbridge-id 122
device(config-rbridge-id-122)# router ospf
device(config-router-ospf-vrf-default-vrf)# bfd
device(config-router-ospf-vrf-default-vrf)# bfd holdover-interval 12
```

Configuring BFD for OSPFv2 globally in a nondefault VRF instance

BFD for OSPFv2 can be configured globally on interfaces where BFD has been configured in nondefault VRF instances.

BFD must be configured using the **ip ospf bfd** command on each OSPFv2 interface to initiate BFD sessions.

1. Enter the **configure terminal** command to access global configuration mode.

   ```
   device# configure terminal
   ```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

   ```
   device(config)# rbridge-id 122
   ```

3. Enter the **router ospf** command and specify a VRF name to enable OSPFv2 in a nondefault VRF instance.

   ```
   device(config-rbridge-id-122)# router ospf vrf red
   ```

4. Enter the **bfd** command to enable BFD.

   ```
   device(config-router-ospf-vrf-red)# bfd
   ```

5. Enter the **bfd holdover-interval** command to set the BFD holdover interval.

   ```
   device(config-router-ospf-vrf-red)# bfd holdover-interval 12
   ```

The following example enables BFD globally and sets the BFD holdover interval to 12 for VRF “red”.

```
device# configure terminal
device(config)# rbridge-id 122
device(config-rbridge-id-122)# router ospf vrf red
device(config-router-ospf-vrf-red)# bfd
device(config-router-ospf-vrf-red)# bfd holdover-interval 12
```

Enabling BFD on a specified OSPFv3-enabled interface

BFD sessions can be configured on one or more OSPFv3-enabled interfaces.

BFD sessions are initiated on specified OSPFv3-enabled interfaces only if BFD is enabled globally using the **bfd** command in OSPFv3 VRF configuration mode.

1. Enter the **configure terminal** command to access global configuration mode.

   ```
   device# configure terminal
   ```

2. Enter the **interface** command and specify an interface.

   ```
   device(config)# interface ve 24
   ```

3. Enter the **ipv6 ospf bfd** command to enable BFD on the specified OSPFv3-enabled interface.

   ```
   device(config-ve-24)# ipv6 ospf bfd
   ```

The following example enables BFD on an OSPFv3-enabled virtual Ethernet (VE) interface.

```
device# configure terminal
device(config)# interface ve 24
device(config-ve-24)# ipv6 ospf bfd
```

Configuring BFD for OSPFv3 globally

BFD for OSPFv3 can be configured globally on interfaces where BFD has been configured.

BFD must be configured using the **ipv6 ospf bfd** command on each OSPFv3 interface to initiate BFD sessions.

1. Enter the **configure terminal** command to access global configuration mode.

   ```
   device# configure terminal
   ```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

   ```
   device(config)# rbridge-id 122
   ```

3. Enter the **ipv6 router ospf** command to enter OSPFv3 router configuration mode and enable OSPFv3.

   ```
   device(config-rbridge-id-122)# ipv6 router ospf
   ```

4. Enter the **bfd** command to enable BFD globally.

   ```
   device(config-ipv6-router-ospf-vrf-default-vrf)# bfd
   ```

5. Enter the **bfd holdover-interval** command to set the BFD holdover interval.

   ```
   device(config-ipv6-router-ospf-vrf-default-vrf)# bfd holdover-interval 20
   ```

The following example enables BFD and sets the BFD holdover interval to 20.

```
device# configure terminal
device(config)# rbridge-id 122
device(config-rbridge-id-122)# ipv6 router ospf
device(config-ipv6-router-ospf-vrf-default-vrf)# bfd
device(config-ipv6-router-ospf-vrf-default-vrf)# bfd holdover-interval 20
```

Configuring BFD for OSPFv3 globally in a nondefault VRF instance

BFD for OSPFv3 can be configured globally on interfaces where BFD has been configured in nondefault VRF instances.

BFD must be configured using the **ipv6 ospf bfd** command on each OSPFv3 interface to initiate BFD sessions.

1. Enter the **configure terminal** command to access global configuration mode.

   ```
   device# configure terminal
   ```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

   ```
   device(config)# rbridge-id 122
   ```

3. Enter the **ipv6 router ospf** command and specify a VRF name to enable OSPFv3 in a nondefault VRF instance.

   ```
   device(config-rbridge-id-122)# ipv6 router ospf vrf orange
   ```

4. Enter the **bfd** command to enable BFD.

```
device(config-ipv6-router-ospf-vrf-orange)# bfd
```

5. Enter the **bfd holdover-interval** command to set the BFD holdover interval.

```
device(config-ipv6-router-ospf-vrf-orange)# bfd holdover-interval 22
```

The following example enables BFD and sets the BFD holdover interval to 22 for VRF “orange”.

```
device# configure terminal
device(config)# rbridge-id 122
device(config-rbridge-id-122)# ipv6 router ospf vrf orange
device(config-ipv6-router-ospf-vrf-orange)# bfd
device(config-ipv6-router-ospf-vrf-orange)# bfd holdover-interval 22
```

BFD for VXLAN extension tunnels

BFD support for Virtual Extensible LAN (VXLAN) extension tunnels can be configured for diagnostic purposes.

VXLAN extension tunnels facilitate a Layer 3 overlay to extend Layer 2 VLANs across VCS clusters. Two different VCS clusters can be connected by a VXLAN tunnel. A VXLAN tunnel can have its end points in both VCS clusters, with the possibility of multiple Layer 3 ECMP paths (four paths) between the endpoints. BFD has the ability to monitor the tunnel for reachability and quick fault detection.

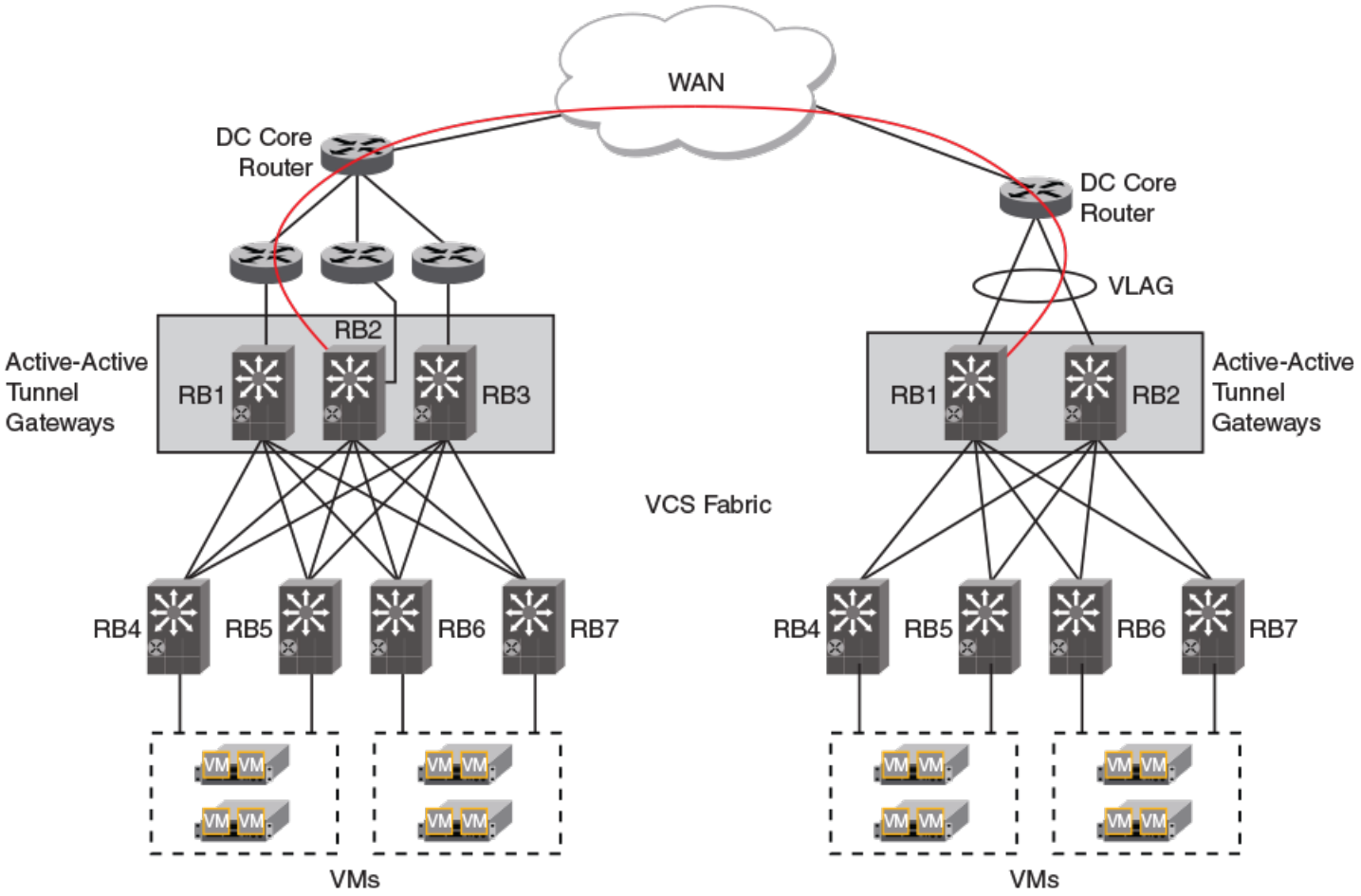
NOTE

RBridges in a VXLAN Network Identifier (VNI) tunnel endpoint (VTEP) are assumed to be connected symmetrically to the destination VTEP. BFD sessions run from all RBridge in the VTEP.

A single BFD session is used to monitor the connectivity of a VXLAN tunnel endpoint. When a tunnel session is created, an RBridge is selected as the “master RBridge”. This master RBridge runs the full BFD state machine for the VXLAN tunnel. BFD session information is also created in other RBridges, but these backup RBridges do not run the full state machine for that specific BFD session. The RBridge with the lowest RBridge ID is selected as the Master RBridge. Furthermore, a newly joined RBridge with a lower RBridge ID than the current master RBridge can pre-empt the mastership from the existing master.

In the figure below, RB1 is the master RBridge for the tunnel session established between the source VTEP and the destination VTEP. RBridges RB2 and RB3 are designated as backup RBridges. The nodes are connected through Inter-Switch Link (ISLs). The participants of the VTEP gateway (RB1, RB2, and RB3) communicate with each other using fabric messages that are transmitted through ISL links.

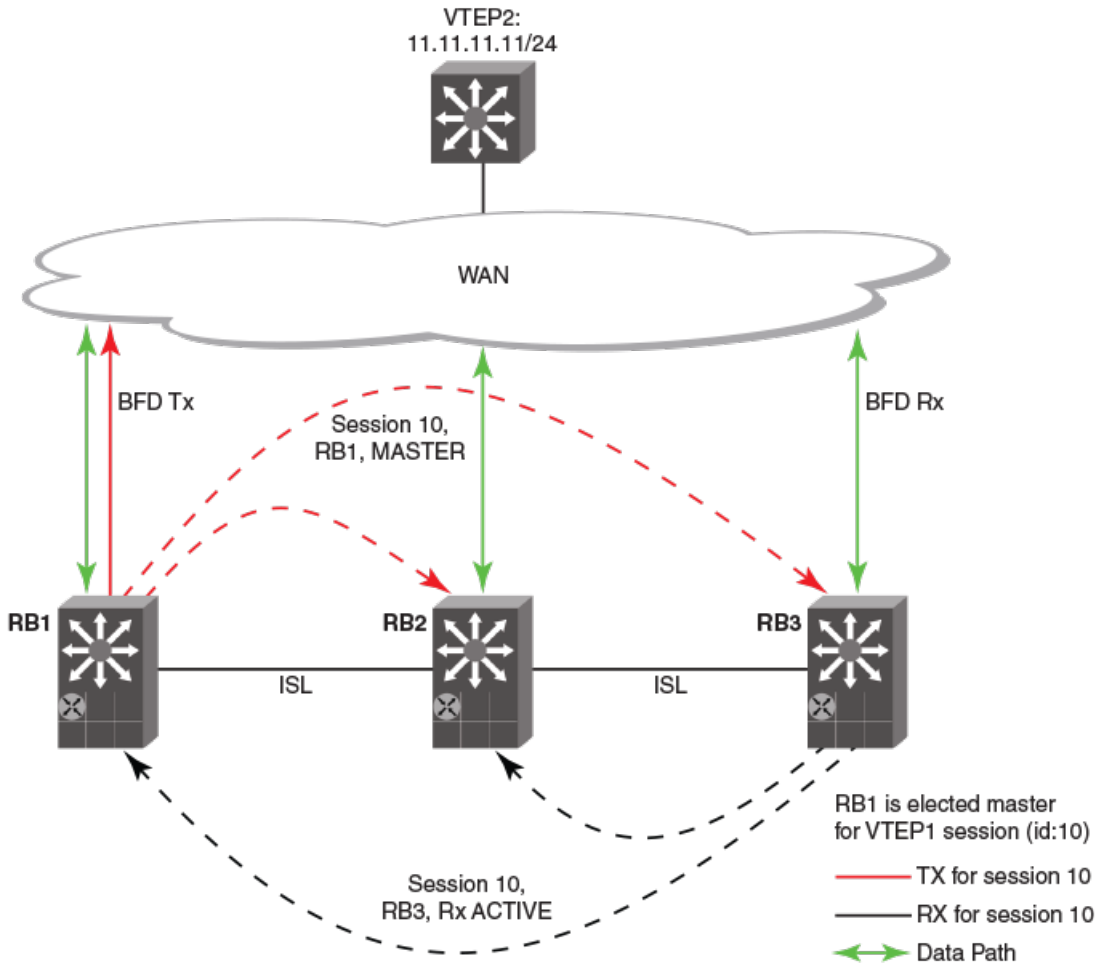
FIGURE 40 BFD for VXLAN extension tunnels



Once a session is established, RB1 continues to transmit BFD control packets based on the negotiated interval. BFD control packets can arrive on any of the gateway RBridges, including the initiator. If control packets become trapped on an RBridge, BFD updates the local forwarding path state to Active, and propagates this information to all RBridges in the cluster.

The figure below shows a BFD session in a cluster where there is at least one communication path between the gateways and the destination host. The session database for all RBridges contains at least one active entry.

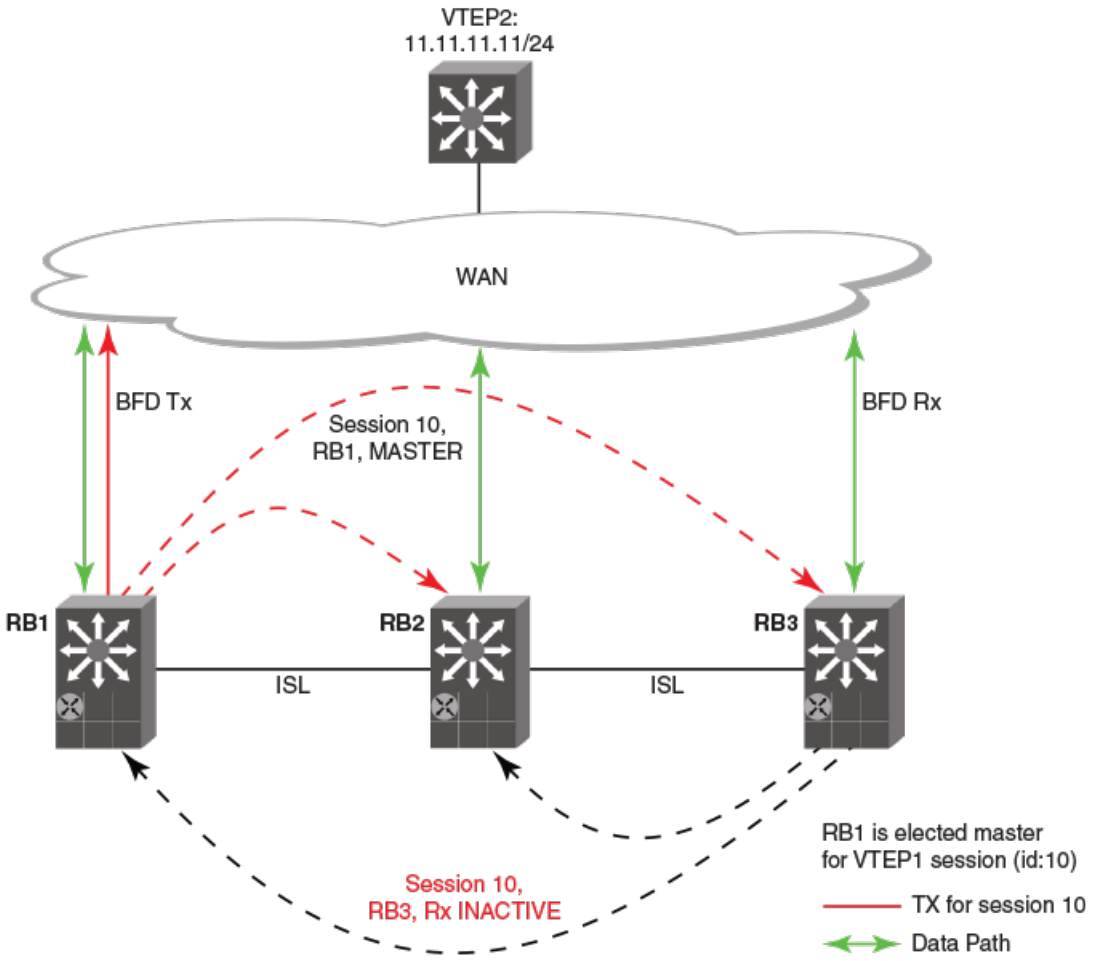
FIGURE 41 Control flow of a BFD Session in an Up state



Session ID	SRC IP	DST IP	Master (Tx)	Rx			Final State
				RB1	RB2	RB3	
10	10.10.10.10/24	10.11.11.11/24	RB1	0	0	1	1

The figure below shows a BFD session in a cluster where all communication paths between the gateways and destination host are lost. The session database for all RBridges contains no active entries.

FIGURE 42 BFD session in a down state



Session ID	SRC IP	DST IP	Master (Tx)	Rx			Final State
				RB1	RB2	RB3	
10	10.10.10.10/24	10.11.11.11/24	RB1	0	0	0	0

A session is declared as down only if the state is declared down on all RBridges. As long as one forwarding path to the destination exists from any of the RBridges in a cluster, a BFD session remains active in the entire fabric.

A maximum of 20 BFD sessions can be configured for extension tunnels.

Configuring BFD on a VXLAN extension tunnel

BFD can be configured on a VXLAN extension tunnel.

1. Enter the **configure terminal** command to access global configuration mode.

```
device# configure terminal
```

2. Enter the **overlay-gateway** command and specify a name to create a VXLAN overlay gateway instance and enter VXLAN overlay gateway configuration mode.

```
device(config)# overlay-gateway gateway1
```

3. Enter the **type** command using the **layer2-extension** keyword to specify that the VXLAN overlay gateway uses a Layer 2 extension.

```
device(config-overlay-gw-gateway1)# type layer2-extension
```

4. Enter the **ip interface** command using the **loopback** keyword to specify an IPv4 loopback interface.

```
device(config-overlay-gw-gateway1)# ip interface loopback 22
```

5. Enter the **site** command and specify a name to create a remote Layer 2 extension site in a VXLAN overlay gateway context and enable VXLAN overlay gateway site configuration mode.

```
device(config-overlay-gw-gateway1)# site s1
```

6. Enter the **ip address** command to specify the destination IPv4 address of the tunnel.

```
device(config-site-s1)# ip address 10.11.12.13
```

7. Enter the **bfd** command to enable BFD on the VXLAN overlay gateway site.

```
device(config-site-s1)# bfd
```

8. Enter the **bfd interval** command with the **min-rx** and **multiplier** keywords to set the BFD session parameters on the VXLAN overlay gateway site.

```
device(config-site-s1)# bfd interval 2000 min-rx 3000 multiplier 26
```

The following example enables BFD on the VXLAN overlay gateway site and sets the BFD session parameters.

```
device# configure terminal
device(config)# overlay-gateway gateway1
device(config-overlay-gw-gateway1)# type layer2-extension
device(config-overlay-gw-gateway1)# ip interface loopback 22
device(config-overlay-gw-gateway1)# site s1
device(config-site-s1)# ip address 10.11.12.13
device(config-site-s1)# bfd
device(config-site-s1)# bfd interval 2000 min-rx 3000 multiplier 26
```

BFD for NSX tunnels

BFD support for NSX tunnels can be configured on the VDX 6740.

An NSX Controller expects a VXLAN Network Identifier (VNI) tunnel endpoint (VTEP) to use BFD to determine the reachability of NSX replicator VTEPs at the end of the tunnels. The VTEP is expected to use this information to determine whether the tunnel is a healthy NSX replicator for sending egress BUM traffic.

In an NSX logical network there can be multiple NSX replicators for redundancy. When there are multiple replicator tunnels, the switch can distribute the BUM traffic across all such tunnels by assigning a different set of VLANs to each of the tunnels. Egress BUM traffic on different VLANs are now forwarded through different tunnels, increasing the overall efficiency of the network. The switch can receive ingress BUM traffic over any VLAN on any NSX replicator tunnel. When a tunnel is deleted or a BFD " down" state is detected, the system automatically reassigns its BUM VLANs to other available replicator tunnels. Refer to the *VXLAN NSX replicator load balancing* section of the*Extreme Network OS Layer 2 Switching Configuration Guide* for more information.

Details of the BUM-enabled NSX replicator tunnel can be printed using the **show tunnel** command. Refer to the *Extreme Network OS Command Reference* for more information on this command.

NOTE
BFD parameters are not configurable in switches for VXLAN
NSX.

For more information on NSX tunnels, refer to the *Extreme Network OS Layer 2 Switching Configuration Guide*.

NOTE
Open vSwitch hardware_vtep v1.3.0 schema support is required on VTEP switches for using BFD in a NSX logical network.

BFD for static routes

Unlike dynamic routing protocols, such as OSPF and BGP, static routing has no method of peer discovery and fault detection. BFD for IPv4 and IPv6 static routes provides rapid detection of failure in the bidirectional forwarding path between BFD peers.

BFD for static routes allows you to detect failures that impact the forwarding path of a static route. This feature supports both single-hop and multihop BFD static routes for both IPv4 and IPv6. Unless the BFD session is up, the gateway for the static route is considered unreachable, and the affected routes are not installed in the routing table. BFD can remove the associated static route from the routing table if the next-hop becomes unreachable indicating that the BFD session has gone down.

Static routes and BFD neighbors are configured separately. A static route is automatically associated with a static BFD neighbor if the static route's next-hop exactly matches the neighbor address of the static BFD neighbor and BFD monitoring is enabled for the static route.

When a static BFD neighbor is configured, BFD asks the routing table manager if there is a route to the BFD neighbor. If a route exists, and if the next-hop is directly connected, BFD initiates a single-hop session. If the next-hop is not directly connected, BFD establishes a multihop session.

When a BFD session goes down because the BFD neighbor is no longer reachable, static routes monitored by BFD are removed from the routing table manager. The removed routes can be added back if the BFD neighbor becomes reachable again. Single-hop BFD sessions use the BFD timeout values configured on the outgoing interface. Timeout values for multihop BFD sessions are specified along with each BFD neighbor. Multiple static routes going to the same BFD neighbor use the same BFD session and timeout values.

BFD considerations and limitations for static routes

There are a number of things to consider when configuring BFD for IPv4 and IPv6 static routes.

- BFD is not supported on interface-based static routes because BFD requires that the next-hop address matches the address of the BFD neighbor.
- Only one static route BFD session for a neighbor is created at any instance. This is always based on the best path for the neighbor.
- Static BFD for a multihop BFD neighbor reachable by way of an Equal-Cost Multi-Path (ECMP) is not supported. Static BFD must be configured explicitly for each next-hop corresponding to each path.
- When an interface goes down, multihop IPv4 static route sessions are not deleted. Multihop IPv6 static route sessions are deleted.
- BFD for static routes is supported in both Local-only mode and Distributed mode.
- BFD sessions can be single-hop or multihop sessions.
- A BFD multihop session is supported for a next-hop resolved through OSPF or BGP.

- If a BFD session goes down and the BFD neighbor had Layer 3 direct connectivity, associated static routes are removed from the routing table so that data packets can use the available alternate path.
- If a BFD neighbor is not directly connected and a BFD session goes down, associated static routes are removed only if an alternate path to the neighbor exists.
- BFD for static routes is supported in both default and nondefault VRFs.
- BFD for IPv6 static routes is supported in both associated and unassociated mode.
- BFD for Link-local IPv6 addresses is supported.
- When configuring BFD for Link-local IPv6 static routes, the source IPv6 address must be link-local and an interface must be provided.

BFD for static routes configuration

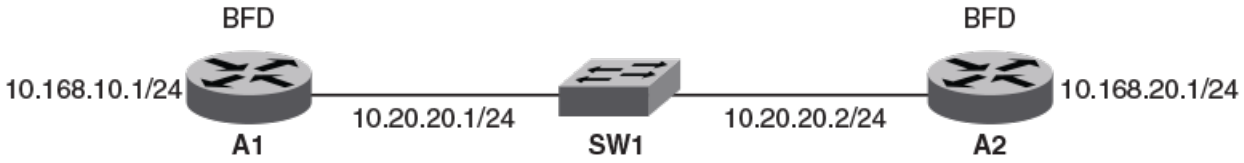
Single-hop BFD IPv4 static route sessions use the timer values configured for the outgoing interface to the directly connected neighbors. Multihop BFD IPv4 static route sessions use the timer values configured using the **ip route static bfd** and **ip route static bfd holdover-interval** commands. If the timer values configured conflict with the timer values set for BGP for the same next-hop, BFD uses the smaller value to meet the more stringent requirement.

Single-hop BFD IPv6 static route sessions use the timer values configured for the outgoing interface to the directly connected neighbors. Multihop BFD IPv6 static route sessions use the timer values configured using the **ipv6 route static bfd** and **ipv6 route static bfd holdover-interval** commands. If the timer values configured conflict with the timer values set for BGP for the same next-hop, BFD uses the smaller value to meet the more stringent requirement.

If you remove a static BFD session, the corresponding session is removed by BFD without removing static routes from the routing table and ongoing traffic is not disrupted. If a BFD session goes down because a BFD neighbor is no longer reachable, all associated static routes are removed from the routing table. Existing traffic on these static routes is interrupted.

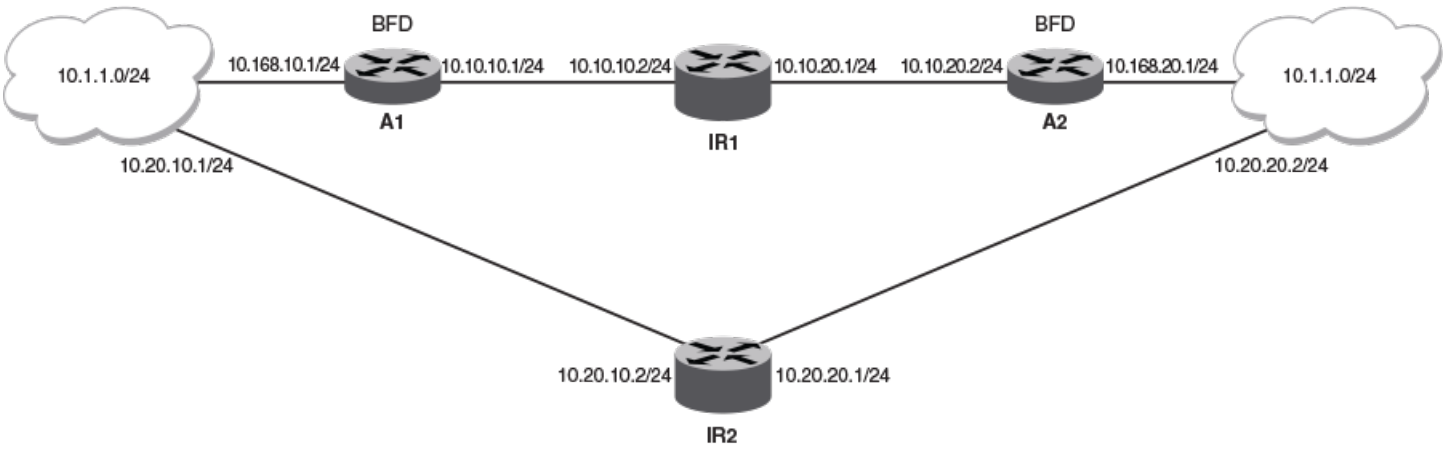
The following figure shows a single-hop static BFD session. A1 has a static route to 10.168.20.0/24 with the next-hop as 10.20.20.2. A2 has a static route to 10.168.10.0/24 with the next-hop as 10.20.20.1. A switch is connected between the routers. BFD can be configured to monitor next-hop 10.20.20.2 on A1 and 10.20.20.1 on A2.

FIGURE 43 Single-hop static BFD session



The following figure shows a multihop static BFD session. Static routes are configured on A1 to reach the 10.1.1.0/24 subnet by way of 10.168.20.1. 10.168.20.1 is reachable by way of the intermediate routers IR1 and IR2. A static route is configured on A2 to reach the 10.1.1.0/24 subnet by way of 10.168.10.1. This next-hop is reachable in turn by way of the intermediate routers IR1 and IR2. If one BFD session goes down, the corresponding route is removed from the routing table and the data packets take another path.

FIGURE 44 Multihop ECMP static BFD session



NOTE

When configuring BFD for static routes, static routes are already installed in the routing table and traffic is running on those static routes. When you configure BFD on these static routes, a similar BFD configuration also occurs on BFD neighbors. If BFD session creation fails or a BFD session does not come up, associated static routes are not removed from the routing table; therefore ongoing traffic on these static routes is not interrupted. A BFD session may not be established if a neighbor is busy or if the maximum number of sessions has been reached on the neighbor. Ongoing traffic on installed static routes is not interrupted.

Configuring BFD on an IP static route

BFD can be configured globally on IP static routes. Repeat the steps in this procedure on each BFD neighbor.

1. Enter the **configure terminal** command to access global configuration mode.

```
device# configure terminal
```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

```
device(config)# rbridge-id 122
```

3. Enter the **ip route static bfd** command, specifying a source and destination IP address, and use the **interval**, **min-rx**, and **multiplier** keywords to configure BFD session parameters on an IP static route.

```
device(config-rbridge-id-122)# ip route static bfd 10.0.2.1 10.1.1.1 interval 500 min-rx 500
multiplier 5
```

4. Enter the **ip route static bfd holdover-interval** command and specify an interval to set the BFD holdover interval globally for IP static routes.

```
device(config-rbridge-id-122)# ip route static bfd holdover-interval 15
```

The following example configures BFD session parameters on an IP static route where the destination IP address is 10.0.2.1 and the source IP address is 10.1.1.1. The BFD holdover interval is set globally to 15 for IP static routes.

```
device# configure terminal
device(config)# rbridge-id 122
device(config-rbridge-id-122)#)# ip route static bfd 10.0.2.1 10.1.1.1 interval 500 min-rx 500 multiplier 5
device(config-rbridge-id-122)# ip route static bfd holdover-interval 15
```

Configuring BFD on an IP static route in a nondefault VRF instance

BFD can be configured on IP static routes in a nondefault VRF instance. Repeat the steps in this procedure on each BFD neighbor.

1. Enter the **configure terminal** command to access global configuration mode.

```
device# configure terminal
```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

```
device(config)# rbridge-id 122
```

3. Enter the **vrf** command and specify a VRF name to enter Virtual Routing and Forwarding (VRF) configuration mode.

```
device(config-rbridge-id-122)# vrf green
```

4. Enter the **address-family ipv4 unicast** command to enter IPv4 address-family unicast configuration mode..

```
device(config-vrf-green)# address-family ipv4 unicast
```

5. Enter the **ip route static bfd** command, specifying a source and destination IP address, and use the **interval**, **min-rx**, and **multiplier** keywords to configure BFD session parameters on an IP static route in a nondefault VRF instance.

```
device(vrf-ipv4-unicast)# ip route static bfd 10.0.0.1 10.1.1.2 interval 500 min-rx 500 multiplier 5
```

The following example configures BFD session parameters on an IP static route in a nondefault VRF instance, where the destination IP address is 10.0.0.1 and the source IP address is 10.1.1.2.

```
device# configure terminal
device(config)# rbridge-id 122
device(config-rbridge-id-122)# vrf green
device(config-vrf-green)# address-family ipv4 unicast
device(vrf-ipv4-unicast)# ip route static bfd 10.0.0.1 10.1.1.2 interval 500 min-rx 500 multiplier 5
```

Configuring BFD on an IPv6 static route

BFD can be configured globally on IPv6 static routes. Repeat the steps in this procedure on each BFD neighbor.

1. Enter the **configure terminal** command to access global configuration mode.

```
device# configure terminal
```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

```
device(config)# rbridge-id 122
```

3. Enter the **ipv6 route static bfd** command, specifying a source and destination IPv6 address and an interface type, and use the **interval**, **min-rx**, and **multiplier** keywords to configure BFD session parameters on an IPv6 static route.

```
device(config-rbridge-id-122)# ipv6 route static bfd fe80::a fe80::b ve 20 interval 100 min-rx 100
multiplier 10
```

4. Enter the **ipv6 route static bfd holdover-interval** command and specify an interval to set the BFD holdover interval globally for IPv6 static routes.

```
device(config-rbridge-id-122)# ipv6 route static bfd holdover-interval 25
```

The following example configures BFD session parameters on an IPv6 static route where the destination IPv6 address is fe80::a and the source IPv6 address is fe80::b. A VE interface is specified and the BFD holdover interval is set globally to 25 for IPv6 static routes.

```
device# configure terminal
device(config)# rbridge-id 122
device(config-rbridge-id-122)#)# ipv6 route static bfd fe80::a fe80::b ve 20 interval 100 min-rx 100
multiplier 10
device(config-rbridge-id-122)# ipv6 route static bfd holdover-interval 25
```

Configuring BFD on an IPv6 static route in a nondefault VRF instance

BFD can be configured on IPv6 static routes in a nondefault VRF instance. Repeat the steps in this procedure on each BFD neighbor.

1. Enter the **configure terminal** command to access global configuration mode.

   ```
   device# configure terminal
   ```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

   ```
   device(config)# rbridge-id 122
   ```

3. Enter the **vrf** command and specify a VRF name to enter Virtual Routing and Forwarding (VRF) configuration mode.

   ```
   device(config-rbridge-id-122)# vrf blue
   ```

4. Enter the **address-family ipv6 unicast** command to enter IPv6 address-family unicast configuration mode..

   ```
   device(config-vrf-blue)# address-family ipv6 unicast
   ```

5. Enter the **ipv6 route static bfd** command, specifying a source and destination IPv6 address and an interface type, and use the **interval**, **min-rx**, and **multiplier** keywords to configure BFD session parameters on an IPv6 static route in a nondefault VRF instance.

   ```
   device(vrf-ipv6-unicast)# ipv6 route static bfd fe70::a fe60::b ve 10 interval 1000 min-rx 2000
   multiplier 20
   ```

The following example configures BFD session parameters on an IPv6 static route in a nondefault VRF instance, where the destination IPv6 address is fe80::a and the source IPv6 address is fe80::b. A VE interface is specified.

```
device# configure terminal
device(config)# rbridge-id 122
device(config-rbridge-id-122)# vrf blue
device(config-vrf-blue)# address-family ipv6 unicast
device(vrf-ipv6-unicast)# ipv6 route static bfd fe70::a fe60::b ve 10 interval 1000 min-rx 2000 multiplier
20
```

Displaying BFD information

Various **show** commands verify information about BFD configurations.

Use one or more of the following commands to verify BFD information. The commands do not have to be entered in this order.

1. Enter the **show bfd** command.

```
device# show bfd

Rbridge-id:1
  BFD State: ENABLED, Version: 1
  Supported Protocols: static-ip, tunnel, ospf6, ospf
  All Sessions: Current: 6 Max Allowed: 250 Max Exceeded Count: 0

  Port       MinTx       MinRx       Mult Sessions
  ====       =====       =====       ==== ========
  Te 1/0/9   50          50          3    1
  Te 1/0/10  50          50          3    1
  Tu 61441   1000        1000        3    1
```

   This example output displays BFD information on a stackable switch.

2. Enter the **show bfd** command.

```
device# show bfd
RBridge: 1
BFD State: ENABLED, Version: 1
Supported Protocols: ospf, Tunnel
All Sessions: Current: 4 Max Allowed: 100 Max Exceeded Count: 0
Agent Sessions: Max Allowed on LC: 40 Max Exceeded Count for LCs: 0

LC   Tx/Rx Sessions   LC  Tx/Rx Sessions  LC  Tx/Rx Sessions  LC  Tx/Rx Sessions
1    4/4              2   2/2             3   0/0             4   0/0
5    0/0              6   0/0             7   0/0             8   0/0

BFD Enabled ports count: 2
Port       MinTx    MinRx    Mult   Sessions
Te 1/2/1   100      100      3       2
Tunnel 1   100      100      3       2
```

   This example output displays BFD information on a chassis.

3. Enter the **show bfd neighbors** command.

```
device# show bfd neighbors
OurAddr                    NeighAddr                   State    Int        Rbridge-id
=======                    =========                   =====    ===        ==========
10.10.10.1                 10.10.10.2                  UP       Ve 10      1
fe80::52eb:1aff:fe13:91d   fe80::52eb:1aff:fe13:c5ed   UP       Ve 11      1
12.12.12.1                 12.12.12.2                  UP       Ve 12      1
2001::1                    2001::2                     DOWN     Ve 13      1
```

   This example output displays BFD neighbor information for the default VRF. To obtain information about non-default VRF instances, a non-default VRF instance must be specified. Refer to the *Extreme Network OS Command Reference* for more information.

4. Enter the **show bfd neighbors application** command with the **ospf** and **details** keywords.

```
device# show bfd neighbors application ospf details

OurAddr            NeighAddr        State    Int          Rbridge-id
=======            =========        =====    ===          ==========
5.0.0.1            5.0.0.2          UP       Te 1/0/10    1

  Local    State: UP            Diag: 0              Demand mode: 0    Poll: 0

  Received State: UP            Diag: 0              Demand mode: 0    Poll: 0
  Final: 1
  Local    MinTxInt(ms): 50     MinRxInt(ms): 50     Multiplier: 3
  Received MinTxInt(ms): 50     MinRxInt(ms): 50     Multiplier: 3
  Rx Count: 265660              Tx Count: 275613
  LD/RD:          1/1           Heard from Remote: Y
  Current Registered Protocols: ospf
  Uptime: 0 day 3 hour 50 min 53 sec 400 msec

OurAddr            NeighAddr        State    Int          Rbridge-id
=======            =========        =====    ===          ==========
2.0.0.1            2.0.0.2          UP       Te 1/0/9     1

  Local    State: UP            Diag: 0              Demand mode: 0    Poll: 0

  Received State: UP            Diag: 0              Demand mode: 0    Poll: 0
  Final: 1
  Local    MinTxInt(ms): 50     MinRxInt(ms): 50     Multiplier: 3
  Received MinTxInt(ms): 50     MinRxInt(ms): 50     Multiplier: 3
  Rx Count: 265559              Tx Count: 275550
  LD/RD:          2/3           Heard from Remote: Y
  Current Registered Protocols: ospf
  Uptime: 0 day 3 hour 50 min 50 sec 248 msec
```

This example displays detailed information about neighbor OSPF sessions.

5. Enter the **show bfd neighbors dest-ip** command with the **details** keyword.

```
device# show bfd neighbors dest-ip 1.1.1.6 details
OurAddr            NeighAddr        State    Int          Rbridge-id
=======            =========        =====    ===          ==========
1.1.1.5            1.1.1.6          UP       Ve 5         5

  Local    State: UP            Diag: 0              Demand mode: 0    Poll: 0
  Received State: UP            Diag: 0              Demand mode: 0    Poll: 0    Final: 1
  Local    MinTxInt(ms): 50     MinRxInt(ms): 50     Multiplier: 3
  Received MinTxInt(ms): 50     MinRxInt(ms): 50     Multiplier: 3
  Rx Count: 226354              Tx Count: 234323
  LD/RD:          1/11          Heard from Remote: Y
  Current Registered Protocols: ospf
  Uptime: 0 day 3 hour 4 min 48 sec 248 msec
```

This example shows detailed neighbor information about a destination device.

6. Enter the **show bfd neighbors interface** command with the **ve** and **details** keywords.

```
device# show bfd neighbors interface ve 5 details
OurAddr            NeighAddr        State    Int          Rbridge-id
=======            =========        =====    ===          ==========
1.1.1.5            1.1.1.6          UP       Ve 5         5

  Local    State: UP            Diag: 0              Demand mode: 0    Poll: 0
  Received State: UP            Diag: 0              Demand mode: 0    Poll: 0    Final: 1
  Local    MinTxInt(ms): 50     MinRxInt(ms): 50     Multiplier: 3
  Received MinTxInt(ms): 50     MinRxInt(ms): 50     Multiplier: 3
  Rx Count: 225447              Tx Count: 233383
  LD/RD:          1/11          Heard from Remote: Y
  Current Registered Protocols: ospf
  Uptime: 0 day 3 hour 4 min 0 sec 208 msec
```

This example shows detailed neighbor information about a specified VE interface.

7. Enter the **show bfd neighbors vrf** command and specify a VRF name.

```
device# show bfd neighbors vrf default-vrf

OurAddr            NeighAddr        State    Int          Rbridge-id
=======            =========        =====    ===          ==========

1.1.1.5             1.1.1.6         UP       Ve 5         5
```

This example shows BFD neighbor information for the default VRF.

8. Enter the **show ip bgp neighbors** command.

```
device# show ip bgp neighbors

    Total number of BGP Neighbors: 1
1   IP Address: 100.1.1.2, AS: 100 (IBGP), RouterID: 19.19.19.19, VRF: default-vrf
    State: ESTABLISHED, Time: 0h6m49s, KeepAliveTime: 60, HoldTime: 180
       KeepAliveTimer Expire in 44 seconds, HoldTimer Expire in 136 seconds
    Minimal Route Advertisement Interval: 0 seconds
       RefreshCapability: Received
    Messages:    Open    Update  KeepAlive Notification Refresh-Req
       Sent    : 1       0       9         0            0
       Received: 1       0       8         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 IPV4  unicast  capability
      Peer configured for IPV4 unicast  Routes
    Neighbor ipv6 MPLS Label Capability Negotiation:
    Neighbor AS4 Capability Negotiation:
    Outbound Policy Group:
       ID: 4, Use Count: 1
    BFD:Enabled,BFDSessionState:UP,Multihop:No
        LastBGP-BFDEvent:RX:Up,BGP-BFDError:No Error
        HoldOverTime(sec) Configured:0,Current:0,DownCount:0
    TCP Connection state: ESTABLISHED, flags:00000033 (0,0)
     Maximum segment size: 1460
    TTL check: 0, value: 0, rcvd: 64
       Byte Sent:   216, Received: 197
       Local host:  100.1.1.1, Local  Port: 8177
       Remote host: 100.1.1.2, Remote Port: 179
```

This output shows BGP neighbor information including configured BFD information.

9. Enter the **show ip ospf** command.

```
device# show ip ospf

OSPF Version                 Version 2
Router Id                    19.19.19.19
ASBR Status                  No
ABR Status                   No         (0)
Redistribute Ext Routes from
Initial SPF schedule delay   0          (msecs)
Minimum hold time for SPFs   0          (msecs)
Maximum hold time for SPFs   0          (msecs)
External LSA Counter         0
External LSA Checksum Sum    00000000
Originate New LSA Counter    4
Rx New LSA Counter           5
External LSA Limit           14447047
Database Overflow Interval   0
Database Overflow State :    NOT OVERFLOWED
RFC 1583 Compatibility :     Enabled
Slow neighbor Flap-Action :  Disabled,   timer 300
Nonstop Routing:             Disabled
Graceful Restart:            Disabled,   timer 120
Graceful Restart Helper:     Enabled
BFD:                         Enabled
LDP-SYNC: Not globally enabled
Interfaces with LDP-SYNC enabled: None
```

This output displays OSPF information including configured BFD information.

10. Enter the **show ip ospf** command with the **extensive** keyword.

```
device# show ip ospf neighbor extensive
Number of Neighbors is 1, in FULL state 1
Port      Address        Pri State     Neigh Address   Neigh ID    Ev      Opt Cnt
Ve 10     10.10.10.1     1   FULL/DR   10.10.10.2      2.2.2.2     6        2   0
  Neighbor is known for 0d:00h:01m:34s and up for 0d:00h:00m:55s
  Neighbor BFD State:UP, BFD HoldoverInterval(sec):Configured:2 Current:0
```

This output displays detailed information about all neighbors including configured BFD information.

11. Enter the **show ipv6 ospf neighbor detail** command.

```
device# show ipv6 ospf neighbor detail
Total number of neighbors in all states: 1
Number of neighbors in state Full      : 1

RouterID        Pri State    DR              BDR             Interface     [State]
2.2.2.2           1 Full     2.2.2.2         1.1.1.1         Ve 11         [BDR]
                  Option: 00-00-13    QCount: 0    Timer: 686
BFD State: UP, BFD HoldoverInterval(sec):Configured: 3 Current: 0
```

This output displays detailed OSPF neighbor information including configured BFD information.

12. Enter the **show tunnel replicator** command.

```
device# show tunnel replicator
Tunnel 61441, mode VXLAN, rbridge-ids 1-2
Ifindex 2080436225, Admin state up, Oper state up, BFD up
Overlay gateway "k", ID 1
Source IP 50.50.50.3 ( Ve 555, Vrid 1 ), Vrf default-vrf
Destination IP 20.20.20.251
Active next hops on rbridge 1:
    IP: 20.20.20.251, Vrf: default-vrf
    Egress L3 port: Ve 20, Outer SMAC: 0005.3365.381f
    Outer DMAC: 0050.5682.48e4
    Egress L2 Port: Po 200, Outer ctag: 20, stag:0, Egress mode: Local
    BUM forwarder: yes
Packet count: RX 20701        TX 264784
Byte count  : RX (NA)            TX 62217299
```

This example displays NSX service node information, including configured BFD status.

Virtual Routing and Forwarding

VRF overview

VRF (Virtual Routing and Forwarding) is a technology that controls information flow within a network by isolating the traffic by partitioning the network into different logical VRF domains. This allows a single router or switch to have multiple containers of routing tables or Forwarding Information Bases (FIB) inside it, with one routing table for each VRF instance. This permits a VRF-capable router to function as a group of multiple virtual routers on the same physical router. VRF, in conjunction with virtual private network (VPN) solutions, guarantees privacy of information and isolation of traffic within a logical VRF domain.

A typical implementation of a Virtual Routing and Forwarding instance (referred to as a VRF) are designed to support Border Gateway Protocol (BGP)/Multiprotocol Label Switching (MPLS) VPNs, whereas implementations of VRF-Lite (also referred to as Multi-VRF) typically are much simpler with reduced scalability. The two VRF flavors have a lot in common but differ in the interconnect schemes, routing protocols used over the interconnect, and also in VRF classification mechanisms.

VRF is supported on the VDX 8770, VDX 6740, and VDX 6940 series platforms. The number of supported VRFs is listed in the online help for your physical device.

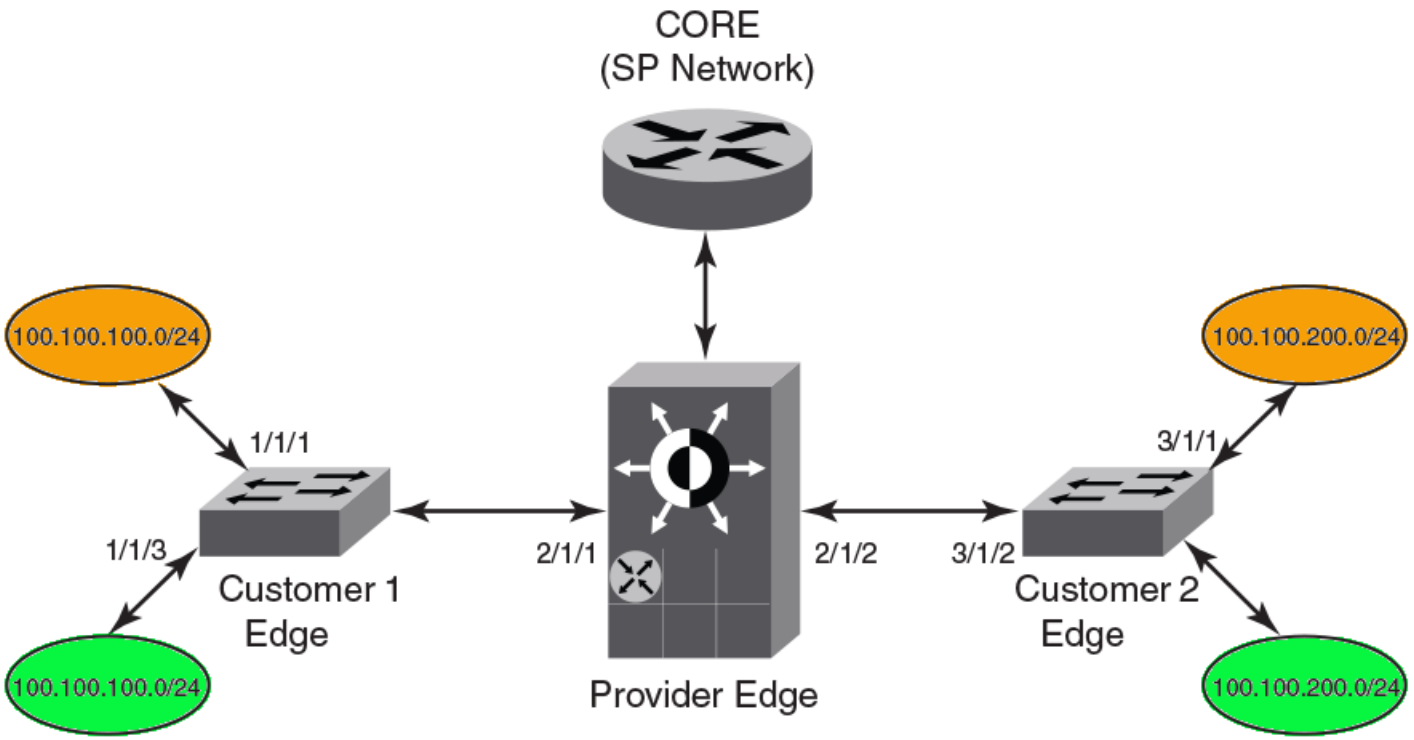
Network OS 4.0.0 and later supports the VRF-Lite implementation. Unless otherwise noted, all references to VRF in this document also apply to VRF-Lite.

VRF topology

This diagram illustrates a typical VRF topology with a single VCS comprising Customer 1 Edge, Provider Edge, and Customer 2 Edge routers.

The orange and green ovals indicate two VPNs supporting two different customer sites. Both of them have overlapping IP subnets; 100.100.100.0/24 and 100.100.200.0/24.

FIGURE 45 VRF topology



Configuring VRF

NOTE
The following configuration gives an example of a typical VRF-Lite use case and is not meant to give an ideal configuration.

The examples in this section are based on the VRF topology on page 353 and use the OSPF routing protocol.

Refer to the *Extreme Network OS Command Reference* for detailed information on VRF commands.

The following example enables routing and configures VRF on the "orange" edge router. If you want to match the VRF topology diagram, you can repeat the steps here to configure the "green" edge router.

1. Enter RBridge ID configuration mode.

```
device(config)# rbridge-id 1
```

2. Set up the VRF instance.
   a) Enter VRF configuration mode and specify "orange" as the VRF name.

```
device(config-rbridge-id-1)# vrf orange
```

   b) Enable IP address-family support for VRF routing.

      The **address-family unicast** command supports both IPv4 and IPv6. This example is based on IPv4. Refer to the *Extreme Network OS Command Reference* for information on IPv6 support.

```
device(config-vrf-orange)# address-family ipv4 unicast
```

   c) Specify the maximum number of routes to be used.

```
device(vrf-oraange-ipv4-unicast)# max-route 3600
```

3. Enable the OSPF routing protocol for the instance in VRF configuration mode, and assign it to area 10.

   NOTE
   All OSPF commands that are present under OSPF router configuration mode are applicable to the new OSPF VRF router configuration mode for a nondefault VRF instance, the same as for the default VRF instance.

```
device(vrf-orange-ipv4-unicast)# router ospf vrf orange
device(config-router-ospf-vrf-orange)# area 10
device(config-router-ospf-vrf-orange)# exit
device(vrf-orange-ipv4-unicast)# exit
```

4. Bind the interface to the VRF instance.

   NOTE
   After a VRF instance is enabled on an interface, all Layer 3 configurations on the interface are removed, and you will need to configure them again, as shown in steps 4 and 5.

```
device(config)# rbridge-id 1
device(config-rbridge-id-1)# interface ve 128
device(config-Ve-128)# vrf forwarding orange
```

5. Configure the static routes.

```
device(vrf-orange-ipv4-unicast)# ip route 10.31.1.0/30 10.30.2.1
```

6. Configure static ARP for the interface.

```
device(vrf-orange-ipv4-unicast)# arp 10.2.2.3 0000.0011.0022 int ve 128
```

7. Confirm the VRF configuration with the **show vrf** command (using **do** in this configuration mode).

```
device(vrf-orange-ipv4-unicast)# do show vrf
Total number of VRFs configured: 4
VrfName                         VrfId  V4-Ucast  V6-Ucast
Red                             3      Enabled     -
default-vrf                     1      Enabled   Enabled
mgmt-vrf                        0      Enabled   Enabled
re                              2      Enabled     -
```

Enabling VRRP for VRF

To enable the Virtual Router Redundancy Protocol (VRRP) or VRRP-Extended (VRRP-E) for a Virtual Routing and Forwarding (VRF) region, an interface should be assigned to a VRF before enabling the VRRP or VRRP-E protocol. The VRRP protocol is enabled or disabled globally on the switch under RBridge ID configuration mode; it cannot be enabled or disabled on a specific VRF instance.

For more information about the VRRP protocol on Network OS switches, see the Configuring VRRPv2 chapter.

1. Enter RBridge ID configuration mode.

```
switch(config)# rbridge-id 1
```

2. Set the protocol to VRRP.

```
switch(config-rbridge-id-1)# protocol vrrp
```

3. Select the interface.

```
switch(config-rbridge-id-1)# interface ve 128
```

4. Add the interface to the VRF.

```
switch(config-Ve-128)# vrf forwarding orange
```

5. Configure an IP address for the interface.

```
switch(config-Ve-128)# ip address 172.128.20.10/24
```

6. Enable the VRRP or VRRP-E protocol for the interface. (In this example, VRRP-E.)

```
switch(config-Ve-128)# vrrp-extended 10
```

7. Set the virtual IP address.

```
switch(config-Ve-128)# virtual-ip 172.128.20.1
```

Inter-VRF route leaking

Virtual Routing and Forwarding (VRF) is a technology that provides you with the ability to have multiple virtual routers on a single physical router or switch. VRFs operate without knowledge of one another unless they are imported or exported into one another using inter-VRF route leaking. This feature allows the leaking of route prefixes from one VRF instance to another VRF instance on the same physical router, which eliminates the need for external routing. This is useful in cases where multiple VRFs share the same path to reach an external domain, while maintaining their internal routing information limited to their own VRF. This feature enables a data center to consolidate multiple VRF services onto a single server.

Both static and dynamic route leaking are supported. Each routed interface (whether virtual or physical) can belong to only one VRF.

Static route leaking provides a mechanism to leak manually configured route entries from a source VRF to a destination VRF.

Dynamic route leaking provides a mechanism to leak routes learned from routing protocols such as BGP and OSPF from a source VRF to a destination VRF. Routes can be leaked by configuring a route map and associating this route map with a source VRF. The match criteria defined in the route map consists of specific route prefixes that exist in the source VRF.

For more information on VRF functionality, refer to VRF overview on page 353.

Dynamic route leak restrictions

- Exporting of route maps is not supported.
- Match criteria in a route map must be provided with prefix lists; other match criteria is ignored.
- Connected routes are not leaked.
- Routes in management-vrf with next-hop as eth0 or management interface are not leaked.

Inter-VRF route conflicts

Inter-VRF route leaking should be deployed only by an advanced user, as route leak configurations in source VRFs may collide with route/interface definitions in target VRFs. This may lead to unpredictable behavior in packet forwarding.

Some of the ways that leaked route conflicts can occur are the following:

- Static route conflict
- Dynamic route conflict
- Connected route conflict

A static route conflict may happen when the same prefix is reachable by two different next hops in the target VRF. The forwarding behavior would be different depending on which command occurred later. This following example presents a static route conflict for 10.1.2.0/24.

```
device(config)# vrf red
device(config-vrf-red)# ip route 10.1.2.0/24 next-hop-vrf green 10.1.1.1
device(config)# vrf green
device(config)# ip route 10.1.2.0/24 18.1.1.1
```

NOTE

However, if the source of the prefix in each case is different (for example, 10.1.2.0 comes from OSPF, BGP, static, or connected, then the method of conflict resolution mentioned above (where the most recent configuration takes precedence) does not apply. The order of preference is as follows: (1) connected, (2) static, (3) OSPF, (4) BGP, assuming that the administrative distances for the prefixes are the defaults. In other words, when the prefix is installed through different sources (OSPF/BGP/static/connected), the prefix with the lowest administrative distance takes precedence. The most recently configured prefix rule applies only if the source of the prefix is the same.

ATTENTION

Ensure that identical prefixes are not leaked.

A dynamic route conflict can occur when dynamic routing protocols advertise different routes to the same prefix in the target VRF.

A connected route conflict is illustrated by the following example:

```
device(config)# vrf red
device(config-vrf-red)# ip route 10.1.2.0/24 next-hop-vrf green 10.1.1.1
device(config)# interface Te 1/2/1
device(config-if-te-1/2/1)# vrf forwarding green
device(config-if-te-1/2/1)# ip address 10.1.2.1/24
```

NOTE

The user must be aware of such possible conflicts before deploying the route leak feature, as currently there is no error checking for these scenarios. A good rule is to make sure that definitions are globally unique and route collisions do not exist.

Displaying Inter-VRF route leaking

The show command for the IP routing table (**show ip route**) displays a '+' sign next to the route type for the leaked routes in a VRF.

The following example shows the static route using the next-hop VRF option for route leaking:

```
device# show ip route
Total number of IP routes: 3
Type Codes - B:BGP D:Connected I:ISIS O:OSPF R:RIP S:Static; Cost - Dist/Metric
BGP  Codes - i:iBGP e:eBGP
ISIS Codes - L1:Level-1 L2:Level-2
OSPF Codes - i:Inter Area 1:External Type 1 2:External Type 2 s:Sham Link
    Destination      Gateway       Port      Cost     Type   Uptime
1   0.0.0.0/0        10.24.64.1    mgmt 1    1/1      S      8m24s
2   1.1.1.0/24       10.1.1.10     Ve 10     1/1      S+     3m11s
3   10.24.64.0/20    DIRECT        mgmt 1    0/0      D      8m28s
```

Notice the '+' sign next to the Type entry for route entry 2.

You can also determine the leaked route for a specific VRF, as part of the (**show ip route**) command, as illustrated in the following example:

```
device# show ip route vrf vrf1
Total number of IP routes: 2
Type Codes - B:BGP D:Connected I:ISIS O:OSPF R:RIP S:Static; Cost - Dist/Metric
BGP  Codes - i:iBGP e:eBGP
ISIS Codes - L1:Level-1 L2:Level-2
OSPF Codes - i:Inter Area 1:External Type 1 2:External Type 2 s:Sham Link
        Destination      Gateway       Port      Cost    Type   Uptime
1       0.0.0.0/0        192.168.64.1  mgmt 1    1/1     S      8m24s
2       10.11.11.0/24    192.168.21.2  Ve 12     1/1     S+     3m11s
```

Notice the '+' sign next to the Type entry for route entry 2.

Configuring static inter-VRF route leaking

Use the following procedure to set up static inter-VRF route leaking.

Refer to the Example of Static Inter-VRF leaking on page 359 for an illustration.

ATTENTION
Static inter-VRF route leaking should be deployed only by an advanced user.

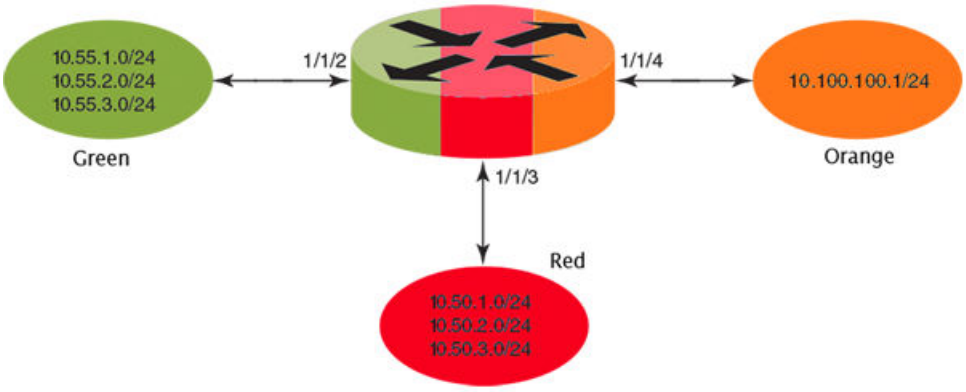
1. Set the switch to RBridge ID configuration mode.
2. Configure the VRF instances you want to be the leaker (source VRF) and where the route is being leaked to (destination VRF).
3. Specify the interface for the source VRF and map it to the source VRF.
4. Enter the IP address/mask to be used for this VRF instance.
5. Specify the interface you want to be the destination VRF and map it to the destination VRF.
6. Specify the IP address/mask to receive the leak.
7. Change the configuration mode to source VRF address-family context.
8. Configure the route to be leaked, specifying the route prefix, the next-hop-VRF name as destination VRF and the next hop to the destination VRF.
9. Optional: (Optional) For bidirectional static inter-VRF route leaking, repeat the above steps, swapping the source and destination addresses.

Example of Static Inter-VRF leaking

In this example, one of the static routes in the "Red" VRF (10.50.2.0/24) is being allowed to communicate with one in the "Green" VRF (10.55.2.0/24).

The center icon represents a VDX router. The red, green and orange ovals represent virtual partitions (VRFs) in that same router. The Destination VRF is where the route is being leaked to ("Green"), and the Source VRF is where the route is being leaked from ("Red").

FIGURE 46 Static Inter-VRF leaking



1. Configure VRF "Green".

```
device(config)# rbridge-id 1
device(config-rbridge-id-1)# vrf Green
device(config-vrf-Green)# address-family ipv4 unicast
```

2. Configure VRF "Red".

```
device(config)# rbridge-id 1
device(config-rbridge-id-1)# vrf Red
device(config-vrf-Red)# address-family ipv4 unicast
```

3. Configure interface in the destination VRF "Green" (using the IP address and subnet mask).

```
device(config)# interface tengigabitethernet 1/1/2
device(config-if-te-1/1/2)# vrf forwarding Green
device(config-if-te-1/1/2)# ip address 10.55.1.2/24
```

4. Configure interface in the source VRF "Red" (using the IP address and subnet mask).

```
device(config)# interface tengigabitethernet 1/1/3
device(config-if-te-1/1/3)# vrf forwarding Red
device(config-if-te-1/1/3)# ip address 10.50.1.2/24
```

5. Configure the source VRF address-family context for static route leaking.

```
device(config)# rbridge-id 1
device(config-rbridge-id-1)# vrf Red
device(config-vrf-Red)# address-family ipv4 unicast
```

6. Configure route leaking for a network (using the IP address and subnet mask), by specifying the destination next-hop VRF name and the next hop in the destination VRF.

   NOTE
   The destination VRF can also be a specific port on an Ethernet interface. Refer to the *Network OS Command Reference* for details on the **ip route next-hop-vrf** command.

   ```
   device(vrf-Red-ipv4-unicast)# ip route 10.55.2.0/24 next-hop-vrf Green 10.55.1.1
   ```

7. Configure route leaking for the default VRF for a network (using the IP address and subnet mask), by specifying the destination next-hop VRF name and the default-vrf in the destination VRF.

   ```
   device(vrf-Red-ipv4-unicast)# ip route 20.0.0.0/24  next-hop-vrf default-vrf 10.1.1.1
   ```

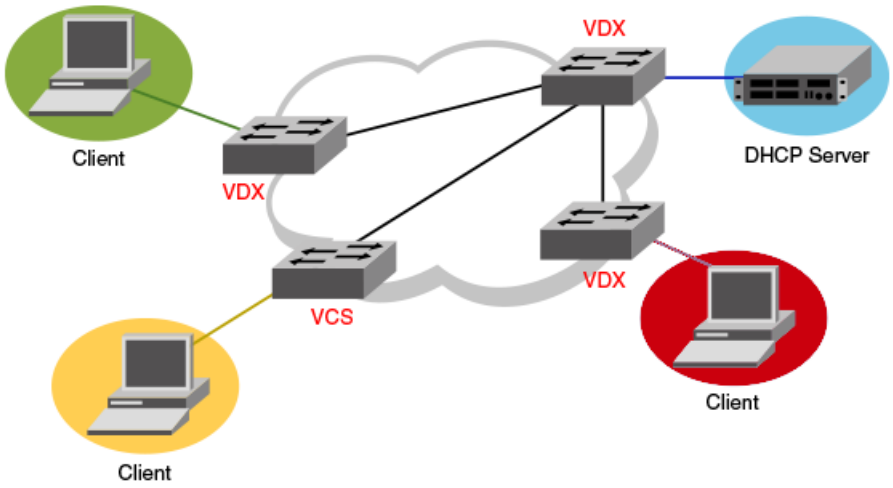
8. (Optional) For bidirectional route leak traffic, you can also configure route leaking from VRF "Green" to VRF "Red".

Inter-VRF route leaking and DHCP relay

In a DHCP relay setting, route leaking is controlled through a single DHCP server (which may be on a different VRF); this permits multiple VRFs to communicate with that server, something that would normally be not permitted. DHCP relay deployments in a data center can use inter-VRF route leaking to achieve server consolidation; this permits clients in multiple VRFs to communicate with a single DHCP server in a different VRF (normally this is not permitted, as VRFs provide route/traffic isolation).

The illustration below shows four VRFs, with three of them connecting to the fourth for DHCP services. For more information on working with DHCP IP Relay, refer to Configuring IP DHCP Relay on page 91.

FIGURE 47 Inter-VRF route leaking example for connecting clients to a DHCP server in a different VRF



The following example shows setting up Inter-VRF route leaking and DHCP between VRF "red" and VRF "blue".

NOTE
Inter-VRF route leaking supports IPv4 and IPv6. Use the **ip address** and **ip route** commands for IPv4 and **ipv6 address** and **ipv6 route** commands for IPv6. These commands support IP addresses, Ethernet interfaces, port channels, and virtual Ethernet (VE)interfaces for the leak destination. Refer to the *Network OS Command Reference*.

1. Set up the VRF forwarding.

```
device(config)# interface ve 100
device(conf-Ve-100)# no shutdown
device(conf-Ve-100)# vrf forwarding red
     <- interface is in VRF "red" ->
device(conf-Ve-100)# ip address 10.1.1.1/24
device(conf-Ve-100)# ip dhcp relay address 20.1.1.2 use-vrf blue
     <- server is in VRF "blue" ->
```

2. Configure the leaked route on VRF "red".

```
device(config) rbridge-id 1
device(config-rbridge-id-1)# vrf red
device(config-vrf-red)# address-family ipv4 max-route
device(vrf-red-ipv4-unicast)#ip route 20.1.1.2/32 next-hop-vrf blue 20.2.1.2
```

Configuring dynamic inter-VRF route leaking

Use the following procedure to configure dynamic inter-VRF route leaking.

Refer to the Example of Dynamic Inter-VRF route leaking on page 362 for an illustration.

ATTENTION
Dynamic inter-VRF route leaking is a feature that should be deployed only by an advanced user.

NOTE
Note the following limitations and considerations for route leaking:

- Leaked routes will not be leaked again.
- Control plane protocols cannot run on leaked routes.
- Leaking the same prefix across VRFs is not supported. That is, a given prefix can be present in multiple VRFs, but it should not be leaked from one VRF to another. The behavior in such a case will be inconsistent.

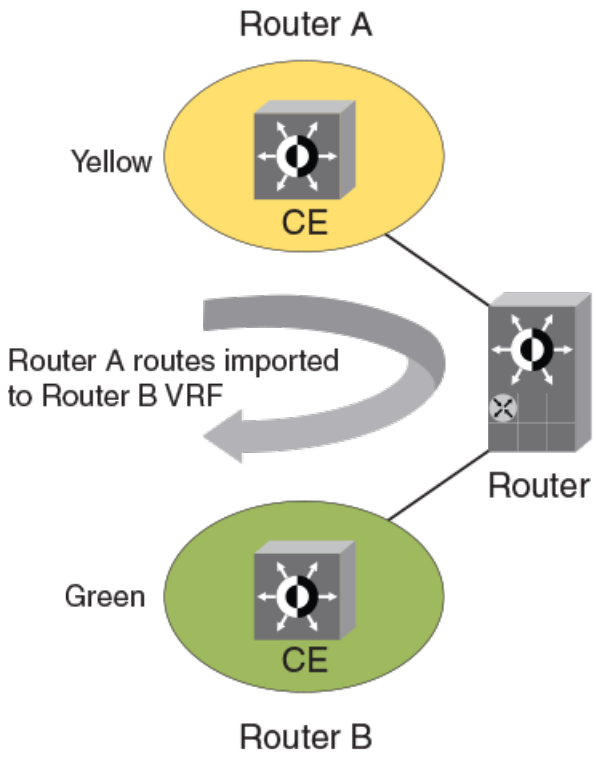
1. Set the switch to RBridge ID configuration mode.
2. Configure the VRF instances you want to be the leaker (source VRF) and where the route is being leaked to (destination VRF).
3. Specify the interface for the source VRF and map it to the source VRF.
4. Enter the IP address/mask to use for this VRF instance.
5. Specify the interface you want to be the destination VRF and map it to the destination VRF.
6. Specify the IP address/mask to receive the leak.
7. Configure the route map and associated prefix-list.
8. Change the configuration mode to destination VRF address-family context. (IPv4 and IPv6 are supported.)
9. Configure the **import** command, specifying the source VRF and route map to be leaked.
10. (Optional) You can leak BGP or OSPF routes that were learned by the source VRF into the destination VRF.

Example of Dynamic Inter-VRF route leaking

In this example, a route map called "import-map" has match criteria specified as prefixes that can be learned by means of routing protocols such as OSPF or BGP.

The figure below depicts a typical dynamic route leak scenario. VRFs “Yellow” and “Green” are virtual partitions in the same router. The Destination VRF is where the route is being leaked to ("Green"), and the Source VRF is where the route is being leaked from ("Yellow"). In this example, IPv4 is used.

FIGURE 48 Dynamic Inter-VRF route leaking



1. Configure VRF "Green".

```
device(config)# rbridge-id 1
device(config-rbridge-id-1)# vrf Green
device(config-vrf-Green)# address-family ipv4 unicast
```

2. Configure VRF "Yellow".

```
device(config)# rbridge-id 1
device(config-rbridge-id-1)# vrf Yellow
device(config-vrf-Yellow)# address-family ipv4 unicast
```

3. Configure an IPv4 prefix list, named "import-prefix" in this example.

```
device(config)# rbridge-id 1
device(config-rbridge-id-1)# ip prefix-list import-prefix permit 10.2.3.0/24
device(config-rbridge-id-1)# ip prefix-list import-prefix permit 10.1.2.0/24
```

4. Configure a route map with "match" conditions.

```
device(config-rbridge-id-1)# route-map import-map permit 10
device(config-route-map-import-map/permit/10)# match ip address prefix-list import-prefix
```

5. Import the desired route map for the specified VRF.

```
device(config)# rbridge-id 1
device(config-rbridge-id-1)# vrf Green
device(config-vrf-Green)# address-family ipv4 unicast
device(vrf-Green-ipv4-unicast)# ip import routes Yellow route-map import-map
```

6. (Optional) You can redistribute any routes learned by OSPF (or BGP) in the source VRF into the destination VRF. The following shows an OSPF example.

```
device(vrf-Green-ipv4-unicast)# exit
device(config-vrf-Green)# exit
device(config-rbridge-id-1)# router ospf
device(config-router-ospf-vrf-default-vrf)# redistribute ospf
```

Commands for dynamic inter-VRF route leaking

Commands you can use to import dynamic routes for inter-VRF route leaking are included in the following table and described in detail in the *Extreme Network OS Command Reference*

TABLE 15 Dynamic inter-VRF route leaking commands

Command	Description	Mode
ip import routes *VRF_name* **route-map** *rmap_name*	Leaks IPv4 routes from a specified VRF to the default VRF, based on match criteria defined in the specified route-map.	RBridge configuration
ip import routes *VRF_name* **route-map** *rmap_name*	Leaks IPv4 routes from one VRF to another VRF, based on match criteria defined in the specified route-map.	VRF address family configuration
ipv6 import routes *VRF_name* **route-map** *rmap_name*	Leaks IPv6 routes from a specified VRF to the default VRF, based on match criteria defined in the specified route-map.	RBridge configuration
ipv6 import routes *VRF_name* **route-map** *rmap_name*	Leaks IPv6 routes from one VRF to another VRF, based on match criteria defined in the specified route-map.	VRF address family configuration
show ip route import	Displays the IPv4 routes imported to a specified VRF.	Privileged EXEC
show ipv6 route import	Displays the IPv6 routes imported to a specified VRF.	Privileged EXEC
redistribute ospf	Redistributes leaked OSPFv3 (or OSPF v2) routes that were imported to another VRF into OSPF of this VRF instance. Refer to the *Extreme Network OS Command Reference* for command options.	OSPF VRF router configuration
redistribute bgp	Redistributes leaked BGP routes that were imported to another VRF into BGP of this VRF instance. Refer to the *Extreme Network OS Command Reference* for command options.	• Address-family ipv4 unicast • Address-family ipv6 unicast • Address-family ipv4 unicast vrf • Address-family ipv6 unicast vrf

NOTE
The **redistribute** commands (introduced in Network OS 6.0.1) enable OSPF or BGP to take the routes leaked from other VRFs and advertise them to peers. This enables the propagation of reachability information to other routers in the network for traffic forwarding.)

Understanding and using management services in default-vrf and mgmt-vrf

The management VRF is a dedicated, secure VRF instance that allows users to manage the router inband on switched virtual interfaces (SVIs) and physical interfaces. The name of this VRF instance is "mgmt-vrf;" this instance cannot be deleted. The default VRF offers access to a subset of management services, whereas all management services are available in Management VRF. It is recommended to check the supportability table below.

A management port is any port that is part of the management VRF. By default, the out-of-band (OOB) management port (the eth0 interface) is part of the management VRF. The OOB port cannot be removed from the management VRF. In addition, Layer 3 virtual and physical ports (also known as front-end or in-band ports) can be part of the management VRF. In-band ports can be moved, by means of the CLI, into and out of the management VRF.

Note the following conditions for the management VRF:

- The management VRF does not support Layer 3 protocols. As a result, dynamic routes are not supported.
- DHCP addresses are preferred over static IP addresses on the management interface. If DHCP is disabled, then the static IP address is reconfigured on the eth0 interface.
- For DHCP gateways, the static gateway (route) has precedence over a dynamic gateway (submitted by DHCP).
- The Virtual Cluster Switching (VCS) virtual IP interface must be in the same subnet as the management IP address (eth0:2).
- The chassis IP address is mapped to the active management module (MM) alias interface eth0:1. The active, standby, and chassis IP addresses must be in the same subnet.
- When the MM IP address is deleted, the IP address on eth0 is also deleted.
- As with other VRFs, the management VRF supports overlapping networks.

The following matrix summarizes the functionality that is supported and unsupported with the management and default VRF. **Y** (Yes) indicates supported; **N** (No) indicates unsupported.

TABLE 16 Supported and unsupported functionality with the management VRF

VRF type	Services	Operational/ debug	IPv4/IPv6	ARP	DHCP relay	DHCP client	Unicast protocols	Multicast protocols	Static routes
	SSH/Telnet - Enabled on mgmt VRF, and Default VRF by default. Access can be further restricted using restrict_ssh script. Per VRF On/Off	FWDL, supportsave	addressing, autoconfig				OSPF, BGP, VRRP, VRRP-E	IGMP, PIM	

TABLE 16 Supported and unsupported functionality with the management VRF (continued)

VRF type	Services	Operational/ debug	IPv4/IPv6	ARP	DHCP relay	DHCP client	Unicast protocols	Multicast protocols	Static routes
	SNMP, NetConf, syslog, sFlow, REST, HTTP - - Not enabled on any VRF by default. Use CLI to turn on these services on mgmt. or default vrf.								
Management									
OOB port	Y	Y	Y	Y (dynamic only)	N	Y	N	N	Y
In-band port	Y	N	Y	Y	N	N	N	N	Y
Default									
Front-end port	Y	N	Y	Y	Y	N	Y	Y	Y
Nondefault									
Front-end port	N	N	Y	Y	Y	N	Y	Y	Y

In addition, vCenter and NSX Controller are supported as follows:

- Management OOB port: **Y**
- Management in-band port: **Y**
- Default (front-end port): **Y**
- Nondefault (front-end port): **N**

Configuring management VRFs

This section provides examples of configuring a management VRF interface and configuring routes on the interface, and disabling a management VRF that has been previously configured on a virtual Ethernet (VE) interface. The management VRF is enabled by means of the **vrf forwarding mgmt-vrf** command, and is disabled by means of the **no** form of that command. You can also configure IPv4 routing by means of the **vrf mgmt-vrf** command in RBridge ID configuration mode.

Enabling a management VRF on an Ethernet interface

The following enables the management VRF on an Ethernet interface and assigns the interface to a subnet.

```
switch(config)# int te 3/0/2
switch(conf-if-te-3/0/2)# vrf forwarding mgmt-vrf
switch(conf-if-te-3/0/2)# ip addr 10.1.1.1/24
```

Disabling a management VRF on a VE interface

The following disables a management VRF previously configured on a VE interface.

```
switch(config)# int ve 100
switch(conf-Ve-100)# no vrf forwarding
```

Configuring IPv4 routing for a management VRF on an RBridge interface

Adding default routes to a management VRF (IPv4 or IPv6)

The following configures an IPv4 route subnet for the management VRF, enters address family IPv4 configuration mode, and assigns the management VRF to an Ethernet interface.

```
switch(config)# rbridge-id 3
switch(confif-rbridge-id-3)# vrf mgmt-vrf
switch(config-vrf-mgmt-vrf)# address-family ipv4 unicast
switch(vrf-ipv4)# ip route 10.1.1.0/32 te 3/0/10
```

The following adds a default IPv4 route to a management VRF.

```
switch(config)# rbridge-id 122
switch(config-rbridge-id-122)# vrf mgmt-vrf
switch(config-vrf-mgmt-vrf)# address-family ipv4 unicast
switch(vrf-ipv4-unicast)# ip route 0.0.0.0/0 10.20.232.1
```

The following adds a default IPv6 route to a management VRF.

```
switch(config)# rbridge-id 122
switch(config-rbridge-id-122)# vrf mgmt-vrf
switch(config-vrf-mgmt-vrf)# address-family ipv6 unicast
switch(vrf-ipv4-unicast)# ipv6 route ::/0 2620:100:0:fa09::1
```

You can confirm the above by using the **show running-config rbridge-id vrf** command, as in the following example. You must enter **mgmt-vrf** manually.

```
sw0# show running-config rbridge-id 122 vrf mgmt-vrf
rbridge-id 122
vrf mgmt-vrf
  address-family ipv4 unicast
   ip route 0.0.0.0/0 10.20.232.1
  !
  address-family ipv6 unicast
   ipv6 route ::/0 2620:100:0:fa09::1
```

Managing management VRFs

There are a variety of show commands that can be used to verify the status of management VRFs, as illustrated in the following examples.

The **show vrf** command indicates the state (A = active) of the management and default VRFs:

```
switch# show vrf

Total number of VRFs configured: 4
VrfName                         VrfId  V4-Ucast  V6-Ucast
Red                             3      Enabled     -
default-vrf                     1      Enabled   Enabled
mgmt-vrf                        0      Enabled   Enabled
re                              2      Enabled     -
```

The **show ip interface** command (with the **do** keyword on an Ethernet interface) indicates that the VRF on this interface is a management VRF:

```
switch(conf-if-te-3/0/10)# do show ip int te 3/0/10

TenGigabitEthernet 3/0/10 is up protocol is up
Primary Internet Address is 10.1.1.1/24 broadcast is 10.1.1.255
IP MTU is 1500
Proxy Arp is Enabled
ICMP unreachables are always sent
ICMP mask replies are never sent
IP fast switching is enabled
Vrf : mgmt-vrf
```

The **show arp vrf mgmt-vrf** command (with the **do** keyword on an Ethernet interface) shows the IP and MAC addresses, the related VE interface, and the status of MAC address resolution:

```
switch(conf-if-te-2/0/9)# do show arp vrf mgmt-vrf

Entries in VRF mgmt-vrf : 1
Address        Mac-address     Interface MacResolved Age       Type
-------------------------------------------------------------------------------
10.1.1.10      0010.9400.0001  Ve 100    yes         00:00:03  Dynamic
```

The **show ip route vrf mgmt-vrf** command indicates which networks are part of the management VRF ("mgmt 1"):

```
switch# show ip route vrf mgmt-vrf

Total number of IP routes: 5
Type Codes - B:BGP D:Connected I:ISIS O:OSPF R:RIP S:Static; Cost - Dist/Metric
BGP Codes - i:iBGP e:eBGP
ISIS Codes - L1:Level-1 L2:Level-2
OSPF Codes - i:Inter Area 1:External Type 1 2:External Type 2 s:Sham Link

    Destination           Gateway         Port      Cost       Type Uptime

2   0.0.0.0/0             10.24.80.1      mgmt 1    0/0             17m9s
3   10.24.80.0/20         DIRECT          mgmt 1    0/0        D    15m41s
4   10.24.82.75/32        DIRECT          mgmt 1    0/0        D    15m41s
5   30.1.1.0/24           DIRECT          Te 3/0/10 0/0        D    0m11
```

Multi-VRF

Multi-VRF overview

Virtual Routing and Forwarding (VRF) allows routers to maintain multiple routing tables and forwarding tables on the same router. A Multi-VRF router can run multiple instances of routing protocols with a neighboring router with overlapping address spaces configured on different VRF instances.

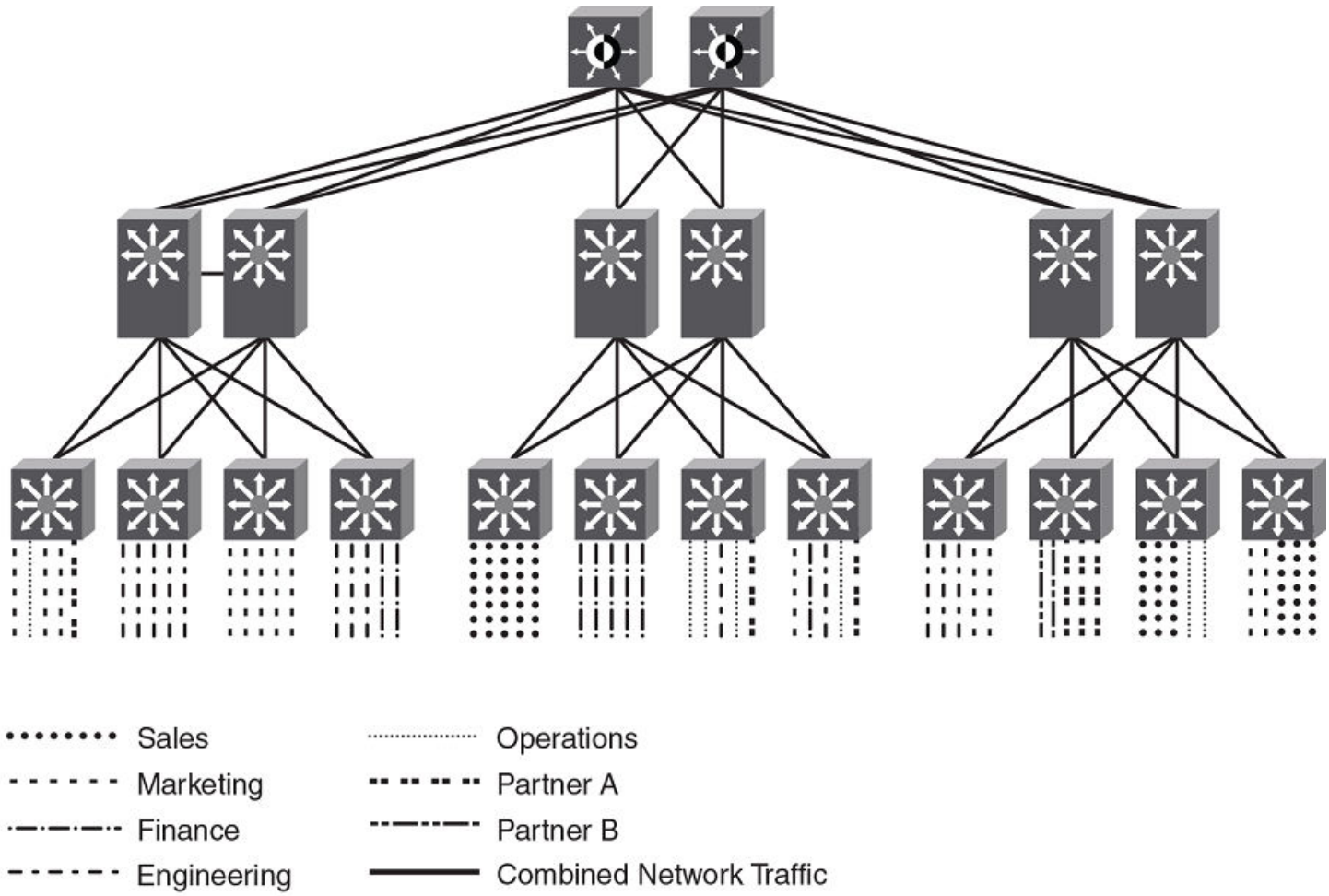
Central to VRF-Lite is the ability to maintain multiple VRF tables on the same Provider Edge (PE) Router. VRF-Lite uses multiple instances of a routing protocol such as OSPF or BGP to exchange route information for a VPN among peer PE routers. The VRF-Lite capable PE router maps an input customer interface to a unique VPN instance. The router maintains a different VRF table for each VPN instance on that PE router. Multiple input interfaces may also be associated with the same VRF on the router, if they connect to sites belonging to the same VPN. This input interface can be a physical interface or a virtual Ethernet interface on a port.

In Multi-VRF deployments:

- Two VRF-capable routers must be directly connected at Layer 3, deploying BGP, OSPF, or static routes.
- Each VRF maintains unique routing and forwarding tables.
- Each VRF can be assigned one or more Layer 3 interfaces on a router to be part of the VRF.
- Each VRF can be configured with IPv4 address family, IPv6 unicast address family, or both.
- A packet's VRF instance is determined based on the VRF index of the interface on which the packet is received.
- Separate routing protocol instances are required for each VRF instance.
- Overlapping address spaces can be configured on different VRF instances.

Multi-VRF deployments provide the flexibility to maintain multiple virtual routers, which are segregated for each VRF instance. The following illustrates a generic, high-level topology where different enterprise functions are assigned unique VRF instances.

FIGURE 49 Example high-level Multi-VRF topology



A Multi-VRF instance can be configured on any of the following:

- Virtual interfaces
- Loopback interfaces
- Ethernet interfaces
- Tunnel interfaces - The tunnel can belong to any user-defined VRF, but the tunnel source and tunnel destination are restricted to the default VRF.

A Multi-VRF instance **cannot** be configured on any of the following:

- Physical interfaces
- Management interfaces

To configure Multi-VRF, perform the following steps:

- Configure VRF instances.
- (Optional) Configure a Route Distinguisher (RD) for new VRF instances.
- Configure an IPv4 address family or IPv6 unicast address family (AF) for new VRF instances.
- Configure routing protocols for new Multi-VRF instances.

- Assign VRF instances to Layer 3 interfaces.

  NOTE
  For the details of Multi-VRF in a BGP EVPN context, refer to "Multi-VRF for BGP EVPN" in the "BGP EVPN" chapter in this document.

In addition, VRFs operate without knowledge of one another unless they are imported or exported into one another by means of inter-VRF route leaking. For details and configuration examples, refer to "Inter-VRF route leaking" in the "VRF" chapter in the *Network OS Layer 3 Routing Configuration Guide*.

Configuring Multi-VRF

Configuring a VRF instance

Do the following to configure a VRF instance.

A device can be configured with more than one VRF instance. You should define each VRF instance before assigning the VRF to a Layer 3 interface. The range of the instance name is from 1 through 255 alphanumeric characters. Each VRF instance is identified by a unique Route Distinguisher (RD), which is prepended to the address being advertised. Because the RD provides overlapping client address space with a unique identifier, the same IP address can be used for different VRFs without conflict. The RD can be an AS number, followed by a colon (:) and a unique arbitrary number as shown below. Alternatively, it can be a local IP address followed by a colon (:) and a unique arbitrary number, as in "1.1.1.1:100." An optional router ID can also be assigned.

Use the **address-family** command in VRF configuration mode to specify an IPv4 or IPv6 address family. For a specific address family you can also configure static route, static ARP, IGMP, and multicast for IPv4, and static route, IPv6 neighbor, and multicast for IPv6.

ATTENTION
Using the **overwrite** option while downloading a configuration from a TFTP server to the running-config will lead to the loss of all VRF configurations when a VRF is configured on a routing interface.

1. Enter global configuration mode.

```
device# configure terminal
```

2. Enter the **rbridge-id** command to specify an RBridge and enter RBridge ID configuration mode.

```
device(config)# rbridge-id 1
```

3. Enter the **vrf** command to create a VRF instance, "corporate" in this example.

```
device(config-rbridge-id-1)# vrf corporate
```

4. (Optional) Assign a Route Distinguisher (RD).

```
device(config-vrf-corporate)# rd 11:11
```

5. (Optional) Assign a router ID.

```
device(config-vrf-corporate)# ip router-id 1.1.1.1
```

6. Use the **address-family unicast (VRF)** command to configure an address family on the VRF and exit. This example uses IPv4.

```
device(config-vrf-corporate)# address-family ipv4 unicast
device(config-vrf-corporate-ipv4)# exit
```

7. Verify the configuration.

```
device(config-vrf-corporate)# do show vrf
Total number of VRFs configured: 2
Status Codes - A:active, D:pending deletion, I:inactive
Name                 Default RD          vrf|v4|v6 Routes Interfaces
corporate            11:11               A  | A| I      0
guest                10:10               A  | A| I      0
Total number of IPv4 unicast route for all non-default VRF is 0
Total number of IPv6 unicast route for all non-default VRF is 0
```

Starting a routing process for a VRF

You must enable a routing protocol for each VRF instance. This example uses OSPF.

1. In global configuration mode, enable OSPF for the VRF instance "corporate."

```
device(config)# router ospf vrf corporate
```

2. Configure the VRF to use OSPF Area 0.

```
device(config-ospf-router-vrf-corporate)# area 0
```

3. (Optional) Configure the VRF to ensure that essential OSPF neighbor state changes are logged, especially in the case of errors.

```
device(config-ospf-router-vrf-corporate)# log adjacency
```

Assigning a Layer 3 interface to a VRF

The following example illustrates how a virtual Ethernet (VE) interface is assigned to a VRF, and how IP addresses and the OSPF protocol are configured.

> ATTENTION
> After you configure a VRF instance on the device, you must assign one or more Layer 3 interfaces (physical or virtual Ethernet) to the VRF. When you do this, all existing IP addresses are deleted; this action also triggers cache deletion, route deletion, and associated cleanup. After you assign an interface to the VRF, you must reconfigure the IP address and interface properties.

1. Enter global configuration mode.

```
device(config)# configure terminal
```

2. Enter the rbridge-id command to specify an RBridge ID and enter RBridge ID configuration mode.

```
device(config)# rbridge-id 1
```

3. Enter the **interface ve** command to specify a virtual Ethernet (VE) interface and enter VE configuration mode.

```
device(config-rbridge-id-1)# interface ve 10
```

4. In VE configuration mode, enable forwarding for the VRF "guest".

```
device(config-Ve-10)# vrf forwarding guest
Warning: All IPv4 and IPv6 addresses (including link-local) on this interface have been removed
have been removed
```

5. Configure an IPv4 address and mask on the VE interface.

```
device(config-Ve-10)# ip address 192.168.1.254/24
```

6. Enable OSPF Area 0.

```
device(config-Ve-10)# ip ospf area 0
```

7. Configure the interface as passive.

```
device(config-Ve-10)# ip ospf passive
```

8. Exit the configuration.

```
device(config-Ve-10)# exit
```

Assigning a loopback interface to a VRF

Do the following to assign a loopback interface to a nondefault VRF.

Because a loopback interface is always available as long as the device is available, it allows routing protocol sessions to stay up even if the outbound interface is down. Assigning a loopback interface to a VRF is similar to assigning any interface. A loopback interface that is not assigned to a nondefault VRF belongs to the default VRF.

1. Enter global configuration mode.

```
device# configure terminal
```

2. Enter the **rbridge-id** command to specify an RBridge and enter RBridge ID configuration mode.

```
device(config)# rbridge-id 1
```

3. Enter the **interface loopback** command to specify a loopback interface and enter interface loopback configuration mode.

```
device(config-rbridge-id-1)# interface loopback 1
```

4. Use the **vrf forwarding** command to assign the interface to the VRF "customer-1" in this example.

```
device(config-Loopbback-1)# vrf forwarding customer-1
```

5. Assign an IPv4 address and mask to the loopback interface.

```
device(config-Loopback-1)# ip address 10.0.0.1/24
```

Verifying a Multi-VRF configuration

The following examples illustrate the use of a variety of show commands that are useful in verifying Multi-VRF configurations.

To verify all configured VRFs in summary mode, enter the **show vrf** command, as in the following example.

```
device# show vrf
Total number of VRFs configured: 2
Status Codes - A:active, D:pending deletion, I:inactive
Name Default RD vrf|v4|v6 Routes Interfaces
green 1:1 A | A| A 12 ve111 ve211 ve311*
red 10:12 A | A| A 4 ve1117 port-id tn1*
Total number of IPv4 unicast route for all non-default VRF is 8
Total number of IPv6 unicast route for all non-default VRF is 8
```

To verify a specific VRF in detail mode, enter the **show vrf detail** *vrf-name* command, as in the following example.

```
device# show vrf green
VRF green, default RD 1:1, Table ID 1
IP Router-Id: 1.1.1.1
Interfaces: ve111 ve211 ve311 ve1116 ve2115
Address Family IPv4
Max Routes: 5500
Number of Unicast Routes: 6
Address Family IPv6
Max Routes: 400
Number of Unicast Routes: 6
```

To verify all configured VRFs in detail mode, enter the **show vrf detail** command, as in the following example.

```
device# show vrf detail
Total number of VRFs configured: 2
VRF green, default RD 1:1, Table ID 1
IP Router-Id: 1.1.1.1
Interfaces: Use "show vrf green" to see the list of interfaces
Address Family IPv4
Max Routes: 5500
Number of Unicast Routes: 6
Address Family IPv6
Max Routes: 400
Number of Unicast Routes: 6
VRF red, default RD 10:12, Table ID 2
IP Router-Id: 1.1.17.1
Interfaces:
Use "show vrf red" to see the list of interfaces
Address Family IPv4
Max Routes: 300
Number of Unicast Routes: 2
Address Family IPv6
Max Routes: 70
Number of Unicast Routes: 2
Total number of IPv4 unicast route for all non-default VRF is 8
Total number of IPv6 unicast route for all non-default VRF is 8
```

The following commands display additional information about a specific application, protocol configuration, or protocol state for both the default VRF and user-defined VRFs.

TABLE 17 Useful show commands

Default VRF	User-defined VRF
show ip route	**show ip route vrf** *vrf-name*
show ip ospf neighbor	**show ip ospf vrf** *vrf-name* **neighbor**
show ip bgp summary	**show ip bgp vrf** *vrf-name* **summary**

Removing a VRF configuration

The following examples illustrate a variety of ways by which you can remove a VRF configuration: deleting a VRF instance from a port, deleting an address family from a VRF, and deleting the VRF globally.

To delete a VRF instance from a specific port, use the **no** form of the **vrf** command. This removes all Layer 3 interface bindings from the VRF, and returns the interface to default VRF mode. All IP addresses and protocol configuration on this Layer 3 interface are removed.

```
device(config-if-te-1/7/1)# no vrf forwarding1
All existing IP and IPv6 address will be removed from port 1/7/1
The port will be returned to default VRF
```

To delete an IPv4 or IPv6 address family from a VRF instance, use the **no** form of the **address-family** command. All configuration related to the address family on all ports of the VRF are removed. Routes allocated to the address family are returned to the global pool.

```
device(config-vrf-customer1)# no address-family ipv4
device(config-vrf-customer1)#
```

To delete a VRF instance globally, use the **no** form of the **vrf** command. All IPv4 or IPv6 addresses are removed from all interfaces.

```
device(config)# no vrf customer1
Warning: All IPv4 and IPv6 addresses (including link-local) from all interfaces in VRF customer1 have been
removed
```

Configuring the maximum number of routes

You can use the **max-route** command to specify the number of routes held in the routing table per VRF instance, for an IPv4 or IPv6 VRF address family.

If this command is not used, the maximum number of routes is 4294967295. This number does not appear in a running configuration.

1. Enter global configuration mode.

   ```
   device# configure terminal
   ```

2. Use the **rbridge-id** command to specify an RBridge ID and enter RBridge ID configuration mode.

   ```
   device(config)# rbridge-id 1
   ```

3. Specify a VRF instance (in this example, "myvrf") and enter VRF configuration mode.

   ```
   device(config-rbridge-id-1)# vrf myvrf
   device(config-vrf-myvrf)#
   ```

4. Enter the **address-family unicast** command, in this example for IPv4, and enter VRF address-family IPv4 unicast configuration mode.

   ```
   device(config-vrf-myvrf)# address-family ipv4 unicast
   ```

5. Enter the **max-route** command and specify the maximum number of routes to be held in the routing table for this VRF instance, 3600 in this example. (The range is from 1 through 4294967295.)

   ```
   device(vrf-myvrf-ipv4-unicast)# max-route 3600
   ```

Inter-VRF route leaking

VRFs operate without the knowledge of one another unless they are imported from or exported into one another by means of inter-VRF route leaking.

Inter-VRF route leaking allows the leaking of specific route prefixes from one VRF instance to another VRF on the same router, thereby eliminating the need for external routing. This feature is useful in cases where multiple VRFs share the same path to reach an external domain, while maintaining the internal routing information limited to their own VRF.

VRF is normally associated with signaling/MPLS in a service provider context. (Multi-VRF is a non-MPLS segmentation of a single physical Infrastructure into virtual and isolated networks.) Each routed interface (virtual or physical) belongs to exactly one VRF. The VRF route leak feature enables a data center to consolidate multiple VRF services to a single server. Assume Host A and Host B need to communicate, and the hosts have the configurations in the following table.

TABLE 18 Host configurations

Host	Host network	Host port	Host VRF	Host VE (subnet)	VDX port
A	2.2.2.1/24	18	Default	10 (2.2.2.10/24)	21
B	3.3.3.1/24	23	Red	20 (3.3.3.10/24)	22

The following illustrates the configuration, by means of the **ip route next-hop-vrf** command, to enable the VRFs to communicate.

```
ip route 3.3.3.0/24 next-hop-vrf red ve 20
!
vrf red
    address-family ipv4 unicast
 ip route 2.2.2.0/24 next-hop-vrf default-vrf ve 10
!
interface Ve 10
 ip proxy-arp
 ip address 2.2.2.10/24
no shutdown
!
interface Ve 20
vrf forwarding red
 ip proxy-arp
 ip address 3.3.3.10/24
 no shutrdown
!
```

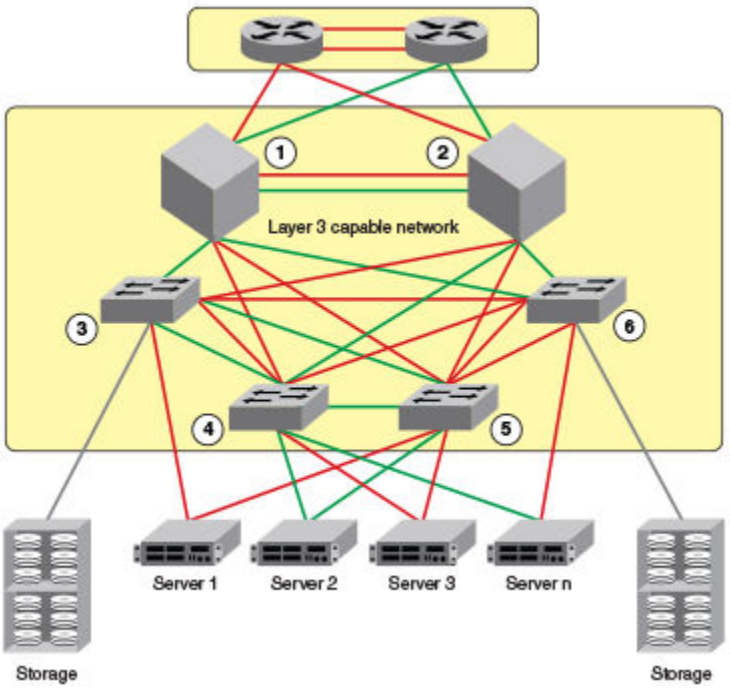
Without local termination support, a ping would be successful from Host A to Host B, but it would fail to reach VE 20 of VDX 152. Similarly, a ping would be successful from Host B to Host A, but it would fail to reach VE10 of VDX 152.

Multi-VRF configuration example

The following is an example of a basic Multi-VRF configuration that uses eBGP with OSPF.

The following example topology shows a typical network that uses the Multi-VRF feature to implement Layer 3 VPNs across two directly connected (at Layer 3) Provider Edge (PE) devices. The Customer Edge (CE) devices can be any router or Layer 3 switch that is capable of running one or many dynamic routing protocols such as BGP, OSPF, or RIP, or even simple static routing. In this example, we use two devices that interconnect all four CE routers with a single link between the two of them.

FIGURE 50 eBGP configured between PE1 and PE2 with OSPF (Area 0) configured between PEs and CEs



1. PE1
2. PE2
3. CE1
4. CE2
5. CE3
6. CE4

Topology details are listed below.

TABLE 19 Topology details

Node	Description	RBridge ID	Networks	Carries routes . . .	Interfaces
PE1	Aggregation (VDX 8770 series)	1	10.1.1.0/24 10.3.1.0/24		1/0/1 1/0/2 1/0/3
PE2	Aggregation (VDX 8770 series)	2	10.2.1.0/24 10.3.1.0/24		2/0/1 2/0/2 2/0/3
CE1	ToR (VDX 6740 series)	3	10.1.1.0/24	10.1.2.0/24 10.1.3.0/24	3/0/1
CE2	ToR (VDX 8770 series)	4	10.1.1.0/24	10.1.2.0/24 10.1.3.0/24	4/0/1
CE3	ToR	5	10.2.1.0/24	10.2.2.0/24	5/0/1

TABLE 19 Topology details (continued)

Node	Description	RBridge ID	Networks	Carries routes . . .	Interfaces
	(VDX 8770 series)			10.2.3.0/24	
CE4	ToR (VDX 6740 series)	6	10.2.1.0/24	10.2.2.0/24 10.2.3.0/24	6/0/1

- Traffic is separated by VRF "green" on VLAN 10 and VRF "red" on VLAN 20.
- eBGP and OSPF (Area 0) are used to connect aggregation switches PE1 and PE2 to a Layer 3-capable aggregation VCS Fabric.
- iBGP and OSPF (Area 0) are used to connect the aggregation switches to the CE switches.
- Alternatively, with only OSPF used, Areas 1 and 2 could carry traffic between the PEs and CEs.

The following configuration examples for PE1, PE2, CE1, CE2, CE3, and CE4 implement the topology above.

> **NOTE**
> The single link between the two PEs could also be replaced by a Layer 2 switched network if direct physical connection between the PEs is not possible. The only requirement for the connections is that the two PEs be directly connected at Layer 3.

In the example topology, because two different VLANs (10 and 20) have overlapping IP address ranges, communication within each customer's VPN across the two PE routers (that is, between CE1 and CE4, and between CE2 and CE3) must be separated by means of two different VRFs ("green" and "red").

> **NOTE**
> The aggregation layer and core must be a Layer 3-capable VCS Fabric.

Multi-VRF with eBGP and OSPF: Configuring PE1

Two VRFs ("red" and "green") are defined, with each having a unique (optional) Route Distinguisher (RD). VRF "green" is assigned an RD value of 10:10, and VRF "red" is assigned an RD value of 20:20.

In the eBGP configuration, PE1 is defined in Local AS 1. VRFs "green" and "red" are configured, and both have the same IP network address assigned (10.3.1.2/24). This is possible because each of the BGP VRF instances has its own BGP tables. This is also the same IP network address that will be assigned to VRFs "green" and "red" on PE2 within Local AS 2. Redistribution of OSPF routes from PE1's CE peers is enabled to all for their advertisement to PE2.

Both VRFs are configured in Area 0 and are directed to redistribute their routes to BGP. The physical interfaces to the CEs are assigned to the appropriate VRF and are configured with the same network address (10.1.1.1/24) and OSPF Area 0.

The virtual Interfaces (Ve 10 and Ve 20) are configured with the same network address (10.3.1.1/24) and for VRF forwarding in the appropriate VRF ("green" or "red").

1. In global configuration mode, create VLANs 10 and 20.

```
device(config)# interface vlan 10
device(config-Vlan-10)# exit
device(config)# interface vlan 20
```

2. From RBridge ID configuration mode, enter interface subtype configuration mode, and then create a virtual Ethernet routing interface for the VLAN.

```
device(config)# rbridge-id 1
device(config-rbridge-id-1)# interface vlan 10
device(config-Vlan-10)# interface ve 10
```

3. Repeat the above steps as appropriate for remaining physical, VLAN, and virtual Ethernet interfaces.
4. Create VRF "green" and assign a Route Distinguisher (optional).

```
device(config)# rbridge-id 1
device(config-rbridge-id-1)# vrf green
device(config-vrf-green)# rd 10:10
device(config-vrf-green)# exit
```

NOTE

This is done for every VRF instance. A Route Distinguisher is optional. Use the **address-family ipv6 unicast** command for IPv6 addresses. Also, you can use the **max-route** command, which helps restrict the maximum number of routes per VRF.

5. Configure VRF "red" and exit the VRF configuration.

```
device(config-rbridge-id-1)# vrf red
device(config-vrf-red)# rd 20:20
device(config-vrf-red)# exit
```

6. In RBridge ID configuration mode, enable BGP routing and configure the following in this IPv4 example.
   a) Enable BGP routing.

```
device(config)# rbridge-id 1
device(config-rbridge-id-1)# router bgp
```

   b) Assign a Local AS number.

```
device(config-bgp-router)# local-as 1
```

   c) Enable IPv4 unicast address-family mode for VRF "green."

```
device(config-bgp-router)# address-family ipv4 unicast vrf green
```

   d) Assign Remote AS 2 as a neighbor with the specified address.

```
device(config-bgp-ipv4u-vrf)# neighbor 10.3.1.2 remote-as 2
```

   e) Assign the appropriate network.

```
device(config-bgp-ipv4u-vrf)# network 10.3.1.0/24
```

   f) Redistribute the OSPF routes into BGP4, specifying the types of routes to be distributed, then exit the address family configuration.

```
device(config-bgp-ipv4u-vrf)# redistribute ospf match internal
device(config-bgp-ipv4u-vrf)# redistribute ospf match external1
device(config-bgp-ipv4u-vrf)# redistribute ospf match external2
device(config-bgp-ipv4u-vrf)# exit
```

7. Repeat as above VRF "red."

```
device(config-rbridge-id-1)# router bgp
device(config-bgp-router)# address-family ipv4 unicast vrf red
device(config-bgp-ipv4u-vrf)# neighbor 10.3.1.2 remote-as 2
device(config-bgp-ipv4u-vrf)# network 10.3.1.0/24
device(config-bgp-ipv4u-vrf)# redistribute ospf match internal
device(config-bgp-ipv4u-vrf)# redistribute ospf match external1
device(config-bgp-ipv4u-vrf)# redistribute ospf match external2
device(config-bgp-ipv4u-vrf)# exit
device(config-bgp-router)# exit
```

8. Enable OSPF routing for VRF "green" and configure the following.
   a) Enable OSPF.

```
device(config)# rbridge-id 1
device(config-rbridge-id-1)# router ospf vrf green
```

   b) Assign Area 0.

```
device(config-ospf-router-vrf-green)# area 0
```

   c) Redistribute the OSPF routes into BGP4 and exit the VRF configuration.

```
device(config-ospf-router-vrf-green)# redistribute bgp
device(config-ospf-router-vrf-green)# exit
device(config-ospf-router)# exit
```

9. Repeat as above for VRF "red".

```
device(config-rbridge-id-1)# router ospf vrf red
device(config-ospf-router-vrf-red)# area 0
device(config-ospf-router-vrf-red)# redistribute bgp
device(config-ospf-router-vrf-red)# exit
```

10. Configure the Ethernet interfaces as appropriate, as in the following example.
    a) Assign an interface to VRF instance "green" and enable forwarding.

```
device(config-rbridge-id-1)# interface tengigabitethernet 1/0/1
device(config-if-te-1/0/1)# vrf forwarding green
```

    b) Assign Area 0.

```
device(config-if-te-1/0/1)# ip ospf area 0
```

    c) Assign an IP network.

```
device(config-if-te-1/0/1)# ip address 10.1.1.1/24
```

    d) Repeat as above for another Ethernet interface and VRF "red" and exit the interface configuration.

```
device(config-if-te-1/0/2)# interface tengigabitethernet 1/0/2
device(config-if-te-1/0/2)# vrf forwarding red
device(config-if-te-1/0/2)# ip ospf area 0
device(config-if-te-1/0/2)# ip address 10.1.1.1/24
device(config-if-te-1/0/2)# exit
```

11. Configure the VE interfaces for the appropriate VRF and network.
    a) Configure VE 10, corresponding to VLAN 10.

```
device(config)# rbridge-id 1
device(config-rbridge-id-1)# interface ve 10
device(config-Ve-10)# vrf forwarding green
device(config-Ve-10)# ip address 10.3.1.1/24
```

    b) Repeat the above for VE 20, corresponding to VLAN 20.

```
device(config-Ve-10)# interface ve 20
device(config-Ve-20)# vrf forwarding red
device(config-Ve-20)# ip address 10.3.1.1/24
device(config-Ve-20# exit
```

Multi-VRF with eBGP and OSPF: Configuring PE2

The PE2 configuration is a mirror image of the PE1 configuration. The only difference is that the BGP neighbor on the corresponding interface has an IP address of 10.3.1.1. This is used in the BGP configuration.

The following summarizes the configuration on PE2.

```
device(config)# rbridge-id 1
device(config-rbridge-id-1)# interface vlan 10
device(config-Vlan-10)# exit
device(config-rbridge-id-1)# interface vlan 20
device(config-Vlan-20)# exit
device(config-rbridge-id-1)# vrf green
device(config-vrf-green)# rd 10:10
device(config-vrf-green)# exit
device(config-rbridge-id-1)# vrf red
device(config-vrf-red)# rd 20:20
device(config-vrf-red)# exit
device(config-rbridge-id-1)# router bgp
device(config-bgp-router)# local-as 1
device(config-bgp-router)# address-family ipv4 unicast vrf green
device(config-bgp-ipv4u-vrf)# neighbor 10.3.1.1 remote-as 2
device(config-bgp-ipv4u-vrf)# network 10.3.1.0/24
device(config-bgp-ipv4u-vrf)# redistribute ospf match internal
device(config-bgp-ipv4u-vrf)# redistribute ospf match external1
device(config-bgp-ipv4u-vrf)# redistribute ospf match external2
device(config-bgp-ipv4u-vrf)# exit
device(config-bgp-router)# address-family ipv4 unicast vrf red
device(config-bgp-ipv4u-vrf)# neighbor 10.3.1.1 remote-as 2
device(config-bgp-ipv4u-vrf)# network 10.3.1.0/24
device(config-bgp-ipv4u-vrf)# redistribute ospf match internal
device(config-bgp-ipv4u-vrf)# redistribute ospf match external1
device(config-bgp-ipv4u-vrf)# redistribute ospf match external2
device(config-bgp-ipv4u-vrf)# exit
device(config-rbridge-id-1)# router ospf vrf green
device(config-ospf-router-vrf-green)# area 0
device(config-ospf-router-vrf-green)# redistribute bgp
device(config-ospf-router-vrf-green)# exit
device(config-rbridge-id-1)# router ospf vrf red
device(config-ospf-router-vrf-red)# area 0
device(config-ospf-router-vrf-red)# redistribute bgp
device(config-ospf-router-vrf-red)# exit
device(config-rbridge-id-1)# interface tengigabitethernet 1/0/2
device(config-if-te-1/0/2)# vrf forwarding green
device(config-if-te-1/0/2)# ip ospf area 0
device(config-if-te-1/0/2)# ip address 10.1.1.1/24
device(config-if-te-1/0/2)# interface tengigabitethernet 1/0/3
device(config-if-te-1/0/3)# vrf forwarding red
device(config-if-te-1/0/3)# ip ospf area 0
device(config-if-te-1/0/3)# ip address 10.1.1.1/24
device(config-if-te-1/0/3)# exit
device(config-rbridge-id-1)# interface ve 10
device(config-Ve-10)# vrf forwarding green
device(config-Ve-10)# ip address 10.3.1.1/24
device(config-Ve-10)# exit
device(config-rbridge-id-1)# interface ve 20
device(config-Ve-20)# vrf forwarding red
device(config-Ve-20)# ip address 10.3.1.1/24
```

Multi-VRF with eBGP and OSPF: Configuring CE1 and CE2

The CE1 and CE2 router configurations are exactly the same. Both are configured in OSPF Area 0 with route redistribution enabled. The IP addresses: 10.1.2.1/32 and 10.1.3.1/32 are configured for the Loopback1 interface allowing them to carry routes from these networks.

1. Enable OSPF routing.

```
device(config)# rbridge-id 1
device(config-rbridge-id-1)# router ospf
device(config-router-ospf-default-vrf)#
```

2. Assign Area 0.

```
device(config-router-ospf-default-vrf)#)# area 0
```

3. Redistribute connected routes into OSPF and exit the OSPF configuration.

```
device(config-router-ospf-default-vrf)# redistribute connected
device(config-router-ospf-default-vrf)# exit
device(config)#
```

4. Configure a loopback interface to support the appropriate networks.

```
device(config_# rbridge-id 1
device(config-rbridge-id-1)# interface loopback 1
device(config-Loopback-1)# ip address 10.1.2.1/32
device(config-Loopback-1)# ip address 10.1.3.1/32
```

5. Configure an Ethernet interface, assign it to Area 0, and assign it to the appropriate network.

```
device(config-Loopback-1)# interface tengigabitethernet 1/0/1
device(config-if-te-1/0/1)# ip ospf area 0
device(config-if-te-1/0/1)# ip address 10.1.1.2/24
```

Multi-VRF with eBGP and OSPF: Configuring CE3 and CE4

The CE3 and CE4 router configurations are exactly the same. Both are configured in OSPF Area 0 with route redistribution enabled. The IP addresses: 10.2.2.1/32 and 10.2.3.1/32 are configured for the Loopback1 interface allowing them to carry routes from these networks.

The following summarizes the configuration.

```
device(config)# rbridge-id 1
device(config-rbridge-id-1)# router ospf
device(config-router-ospf-default-vrf)# area 0
device(config-router-ospf-default-vrf)# redistribute connected
device(config-router-ospf-default-vrf)# exit
device(config-rbridge-id-1)# interface loopback 1
device(config-Loopback-1)# ip address 10.2.2.1/32
device(config-Loopback-1)# ip address 10.2.3.1/32
device(config-Loopback-1)# exit
device(config-rbridge-id-1)# interface tengigabitethernet 1/0/1
device(config-if-te-1/0/1)# ip ospf area 0
device(config-if-te-1/0/1)# ip address 10.2.1.2/24
```

VRRPv2

VRRPv2 overview

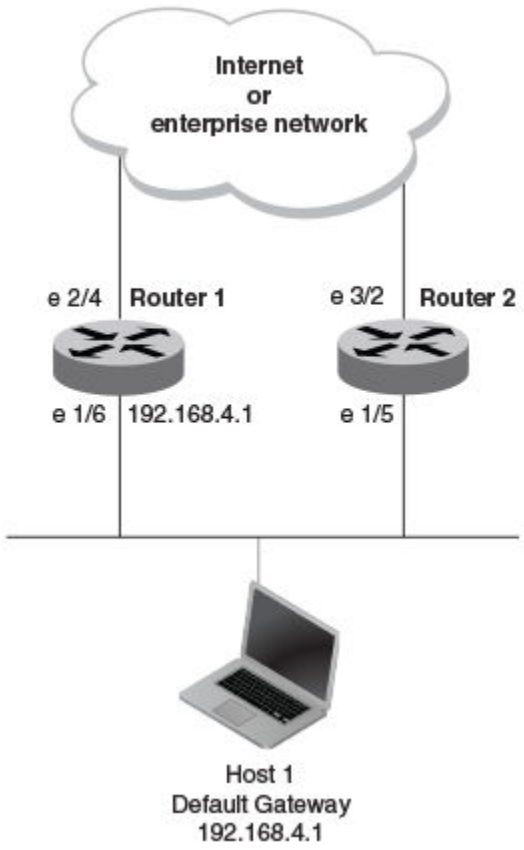
Virtual Router Redundancy Protocol (VRRP) is an election protocol that provides redundancy to routers within a Local Area Network (LAN).

VRRP was designed to eliminate a single point of failure in a static default-route environment by dynamically assigning virtual IP routers to participating hosts. A virtual router is a collection of physical routers whose interfaces must belong to the same IP subnet. A virtual router ID (VRID) is assigned to each virtual router, but there is no restriction against reusing a VRID with a different address mapping on different LANs.

> NOTE
> VRRP extended (VRRP-E) is an extended version of the VRRP protocol. Extreme Networks developed VRRP-E as a proprietary protocol to address some limitations in standards-based VRRP.

Before examining more details about how VRRP works, it is useful to see why VRRP was developed to solve the issue of a single point of failure.

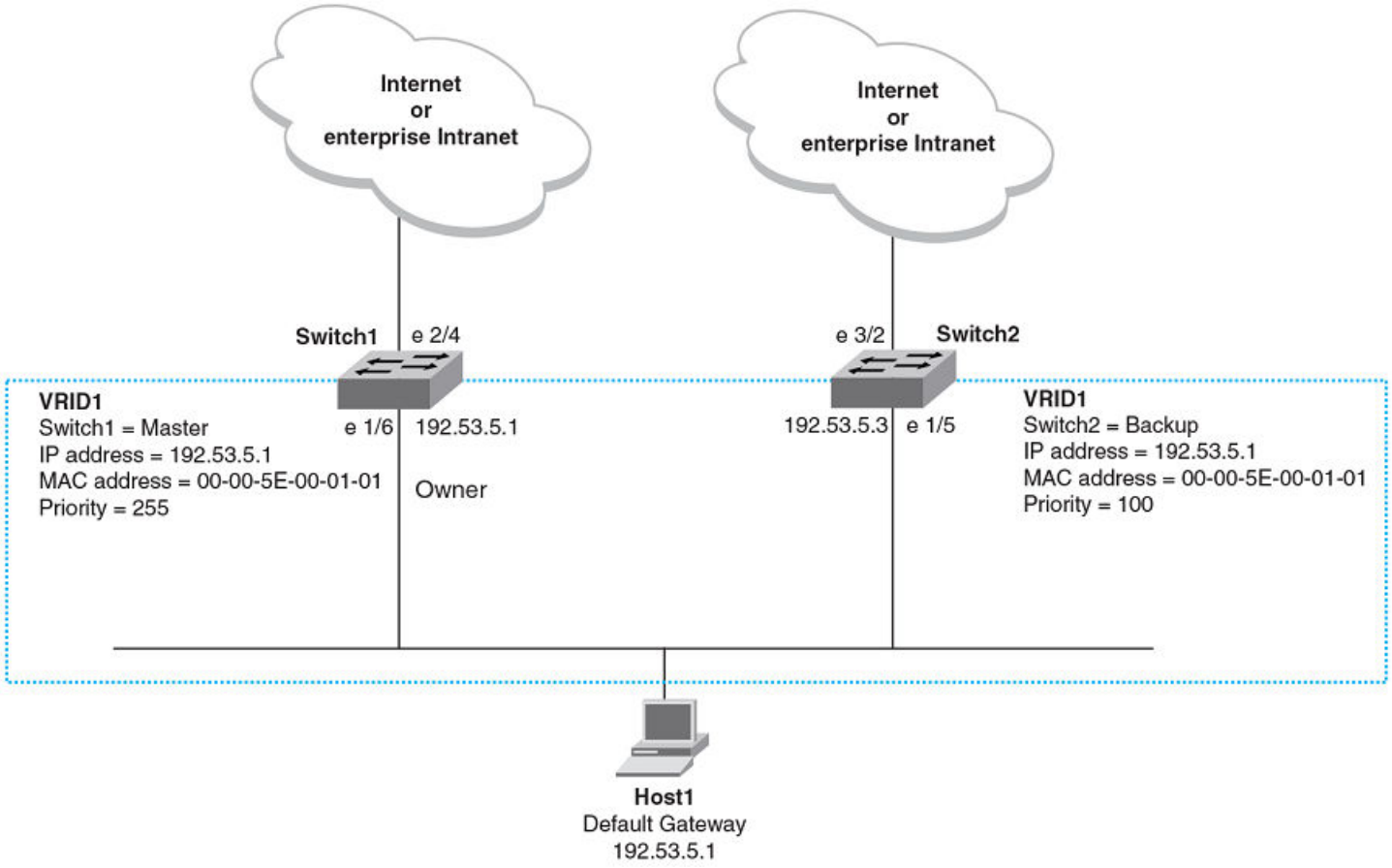
FIGURE 51 Single point of failure with Device 1 being the Host1 default gateway



To connect to the Internet or an internal intranet Host 1, in the figure, uses the IP address of 192.168.4.1 on Router 1 as its default gateway. If this interface goes down, Host 1 is cut off from the rest of the network. Router 1 is a single point of failure for Host 1 to access other networks. In small networks, the administrative burden of configuring Router 2 as the new default gateway is not an issue, but in larger networks reconfiguring default gateways is impractical. Configuring a VRRP virtual router on Router 1 and Router 2 provides a redundant path for the hosts. VRRP allows you to provide alternate router paths for a host without changing the IP address or MAC address by which the host knows its gateway.

To illustrate how VRRP works, the following figure shows the same network, but a VRRP virtual router is configured on the two physical routers, Router 1 and Router 2. This virtual router provides redundant network access for Host 1. If Router 1 were to fail, Router 2 would provide the default gateway out of the subnet.

FIGURE 52 Devices configured as VRRP virtual routers for redundant network access for Host 1



The blue rectangle in the figure represents a VRRP virtual router. When you configure a virtual router, one of the configuration parameters is a group number (also known as a virtual router ID or VRID), which can be a number from 1 through 255. The virtual router is identified with a group, and within the VRRP group, there is one physical device that forwards packets for the virtual router and this is called a master VRRP device. The VRRP master device may be a Layer 3 switch or a router.

In VRRP, one of the physical IP addresses is configured as the IP address of the virtual router, the virtual IP address. The device on which the virtual IP address is assigned becomes the VRRP owner, and this device responds to packets addressed to any of the IP addresses in the virtual router group. The owner device becomes the master VRRP device by default and is assigned the highest priority. Backup devices are configured as members of the virtual router group, and, if the master device goes offline, one of the backup devices assumes the role of the master device.

NOTE

VRRP operation is independent of BGP4 and OSPF. Their operation is unaffected when VRRP is enabled on the same interface as BGP4 or OSPF.

VRRP terminology

Before implementing VRRP in your network, you must understand some key terms and definitions.

The following VRRP-related terms are in logical order, not alphabetic order:

Virtual router	A collection of physical routers that can use VRRP to provide redundancy to routers within a LAN.
Virtual router group	A group of physical routers that are assigned to the same virtual router.
Virtual router address	The virtual router IP address must belong to the same subnet as a real IP address configured on the VRRP interface, and it can be the same as a real IP address configured on the VRRP interface. The virtual router whose virtual IP address is the same as a real IP address is the IP address owner and the default master.
Owner	The owner is the physical router whose real interface IP address is the IP address that you assign to the virtual router. The owner responds to packets addressed to any of the IP addresses in the corresponding virtual router. The owner, by default, is the master and has the highest priority (255).
Master	The physical router that responds to packets addressed to any of the IP addresses in the corresponding virtual router. For VRRP, if the physical router whose real interface IP address is the IP address of the virtual router, then this physical router is always the master.
Backup	Routers that belong to a virtual router, but are not the master. If the master becomes unavailable, the backup router with the highest priority (a configurable value) becomes the new master. By default, routers are given a priority of 100.

VRRPv2 limitations on VDX devices

The implementation of VRRPv2 varies across the VDX products.

Virtual routers must be configured in a Virtual Cluster Switching (VCS) environment.

Only IPv4 support is provided in VRRPv2. VRRPv3 supports both IPv4 and IPv6.

Supported ports:

- For VRRP—fortygigabitethernet, tengigabitethernet, gigabitethernet, port-channel, and ve
- For VRRP-E—ve ports only

System Resource	VDX 87xx	VDX 6740, 6740T, 6740T-1GT	VDX 6940-36Q, 6940-144S
Max # of VRRP and VRRP-E sessions	1024	255	512

The maximum number of virtual IP addresses per virtual router session is 32 for VRRP, and one for VRRP-E.

VRRP hold timer

The hold timer delays the preemption of a master VRRP device by a high-priority backup device.

A hold timer is used when a VRRP-enabled device that was previously a master device failed, but is now back up. This restored device now has a higher priority than the current VRRP master device, and VRRP normally triggers an immediate switchover. In this situation, it is possible that not all software components on the backup device have converged yet. The hold timer can enforce a waiting period before the higher-priority backup device assumes the role of master VRRP device again. The timer must be set to a number greater than 0 seconds for this functionality to take effect.

Hold timer functionality is supported in both version 2 and version 3 of VRRP and VRRP-E.

VRRP interval timers

Various timers for the intervals between hello messages sent between devices running VRRP can be configured.

Hello intervals

Hello messages are sent from the master VRRP device to the backup devices. The purpose of the hello messages is to determine that the master device is still online. If the backup devices stop receiving hello messages for a period of time, as defined by the dead (or master-down-interval) interval, the backup devices assume that the master device is offline. When the master device is offline, the backup device with the highest priority assumes the role of the master device.

Backup hello message state and interval

By default, backup devices do not send hello messages to advertise themselves to the master device. Hello messages from backup devices can be activated, and the messages are sent at 60-second intervals, by default. The interval between the backup hello messages can be modified.

ARP and VRRP control packets

Control packets for ARP and VRRP are handled differently by VRRP and VRRP-E.

Source MAC addresses in VRRP control packets

- VRRP—The virtual MAC address is the source.
- VRRP-E—The physical MAC address is the source.

VRRP control packets

- VRRP—Control packets are IP type 112 (reserved for VRRP), and they are sent to the VRRP multicast address 224.0.0.18.
- VRRP-E—Control packets are UDP packets destined to port 8888, and they are sent to the all-router multicast address 224.0.0.2.

Gratuitous ARP

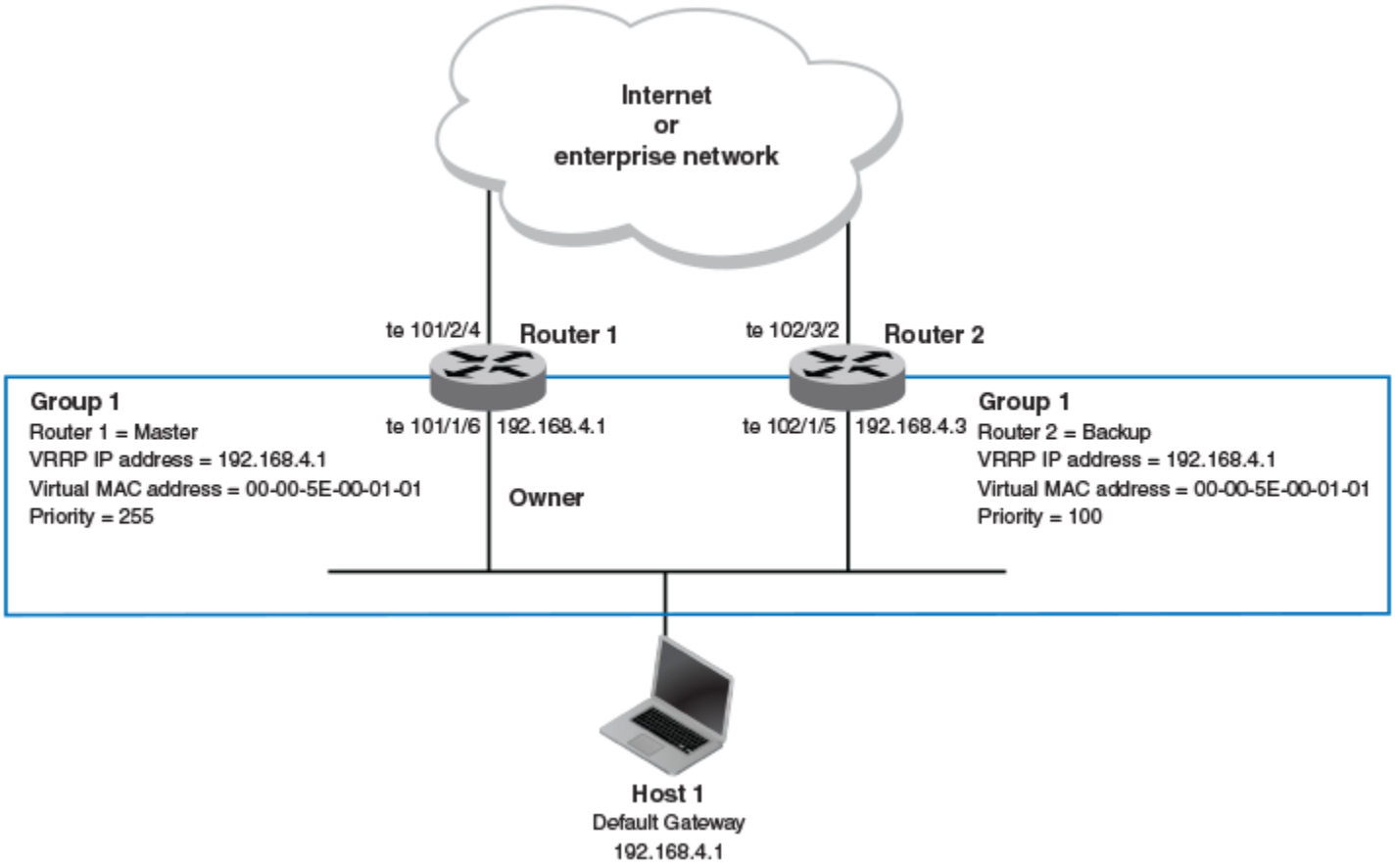
When a VRRP device (either master or backup) sends an ARP request or a reply packet, the MAC address of the sender is the MAC address of the router interface. One exception is if the owner sends an ARP request or a reply packet, in which case the MAC address of the sender is the virtual MAC address. Only the master answers an ARP request for the virtual router IP address. Any backup router that receives this request forwards the request to the master.

- VRRP—A control message is sent only once when the VRRP device assumes the role of the master.
- VRRP-E—A control message is sent every 2 seconds by the VRRP-E master device because VRRP-E control packets do not use the virtual MAC address.

Enabling a master VRRP device

This task is performed on the device that is designated as the master VRRP device. For example, Router 1 is the master VRRP device In the figure that follows.

FIGURE 53 Basic VRRP topology



1. On the device designated as the master VRRP device, and from privileged EXEC mode, enter configuration mode.

```
device# configure terminal
```

2. Enter the **rbridge-id** command, using the RBridge ID (which has an asterisk next to it when you run a **do show vcs** command).

```
device(config)# rbridge-id 101
```

3. Globally enable VRRP.

```
device(config-rbridge-id-101)# protocol vrrp
```

4. Configure the Ethernet interface link for Router 1.

```
device(config-rbridge-id-101)# interface tengigabitethernet 101/1/6
```

5. Configure the IP address of the interface.

```
device(config-if-te-101/1/6)# ip address 192.168.4.1/24
```

6. Assign Router 1 to a group called Group 1.

```
device(config-if-te-101/1/6)# vrrp-group 1
```

NOTE
You can assign a group number in the range of 1 through 255.

7. Assign a virtual router IP address.

```
device(config-vrrp-group-1)# virtual-ip 192.168.4.1
```

NOTE
For VRRP, the physical router whose IP address is the same as the virtual router group IP address becomes the owner and master.

The following example configures a VRRP master device.

```
device# configure
device(config)# rbridge-id 101
device(config-rbridge-id-101)# protocol vrrp
device(config-rbridge-id-101)# interface tengigabitethernet 101/1/6
device(config-if-te-101/1/6)# ip address 192.168.4.1/24
device(config-if-te-101/1/6)# vrrp-group 1
device(config-vrrp-group-1)# virtual-ip 192.168.4.1
```

Enabling a backup VRRP device

This task is performed on a device that is to be designated as a backup VRRP device. For example, Router 2 in Figure 53 on page 388 is assigned as a backup device. Repeat this task for all devices that are to be designated as backup devices.

1. On the device designated as a backup VRRP device, and from privileged EXEC mode, enter global configuration mode.

```
device# configure terminal
```

2. Enter the **rbridge-id** command, using the RBridge ID (which has an asterisk next to it when you run a **do show vcs** command).

```
device(config)# rbridge-id 102
```

3. Globally enable VRRP.

```
device(config-rbridge-id-102)# protocol vrrp
```

4. Configure the Ethernet interface for Router 2.

```
device(config-rbridge-id-102)# interface tengigabitethernet 102/1/5
```

5. Configure the IP address of interface:

```
device(config-if-te-102/1/5)# ip address 192.168.4.3/24
```

NOTE
This router will become the backup router to Router 1.

6. Assign Router 2 to the same VRRP group as Router 1.

```
device(config-if-te-102/1/5)# vrrp-group 1
```

7. To assign Group 1 a virtual IP address, use the same virtual IP address you used for Router 1.

```
device(config-vrrp-group-1)# virtual-ip 192.168.4.1
```

The following example configures a backup VRRP device.

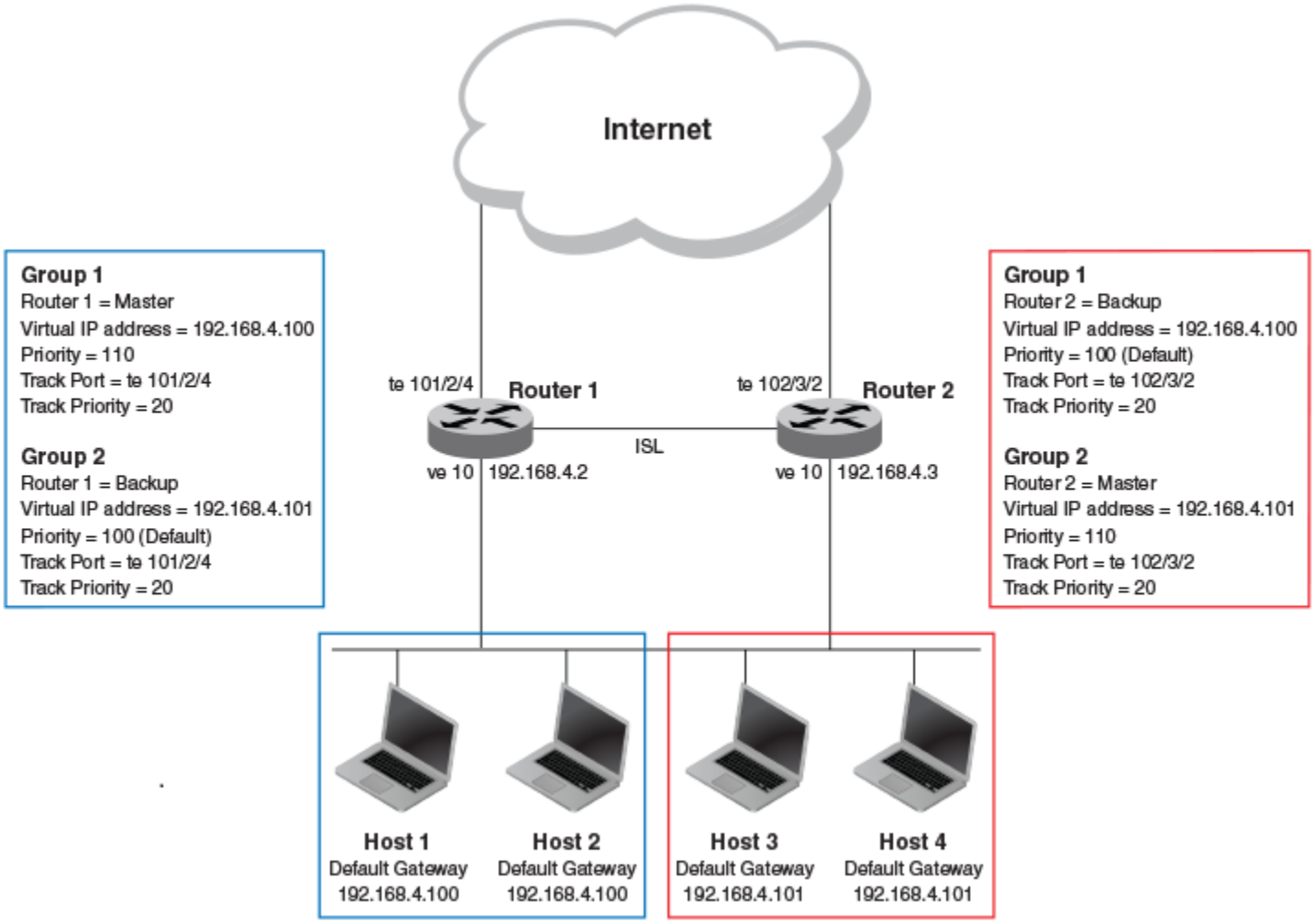
```
device# configure
device(config)# rbridge-id 102
device(config-rbridge-id-102)# protocol vrrp
device(config-rbridge-id-102)# interface tengigabitethernet 102/1/5
device(config-if-te-102/1/5)# ip address 192.168.4.3/24
device(config-if-te-102/1/5)# vrrp-group 1
device(config-vrrp-group-1)# virtual-ip 192.168.4.1
```

VRRP multigroup clusters

Multigroup clusters allow redundancy for host devices and are supported by both VRRP and VRRP-E version 2 and version 3.

The figure below depicts a commonly employed virtual router topology. This topology introduces redundancy by configuring two virtual router groups — the first group has Router 1 as the master and Router 2 as the backup, and the second group has Router 2 as the master and Router 1 as the backup. This type of configuration is sometimes called *Multigroup VRRP*.

FIGURE 54 Two routers configured for dual redundant network access for the host



In this example, Router 1 and Router 2 use VRRP-E to load share as well as provide redundancy to the hosts. The load sharing is accomplished by creating two VRRP-E groups, each with its own virtual IP addresses. Half of the clients point to Group 1's virtual IP address as their default gateway, and the other half point to Group 2's virtual IP address as their default gateway. This enables some of the outbound Internet traffic to go through Router 1 and the rest to go through Router 2.

Router 1 is the master for Group 1 (master priority = 110) and Router 2 is the backup for Group 1 (backup priority = 100). Router 1 and Router 2 both track the uplinks to the Internet. If an uplink failure occurs on Router 1, its backup priority is decremented by 20 (track-port priority = 20) to 90, so that all traffic destined to the Internet is sent through Router 2 instead.

Similarly, Router 2 is the master for Group 2 (master priority = 110) and Router 1 is the backup for Group 2 (backup priority = 100). Router 1 and Router 2 are both tracking the uplinks to the Internet. If an uplink failure occurs on Router 2, its backup priority is decremented by 20 (track-port priority = 20) to 90, so that all traffic destined to the Internet is sent through Router 1 instead.

Configuring multigroup VRRP routing

Configuring VRRP multigroup clusters provides access redundancy to the host devices.

Before configuring this task, ensure that a virtual LAN, named vlan 10, has been created.

To implement the configuration of VRRP multigroup clusters as shown in Figure 54 on page 391, configure one VRRP-E router to act as a master in the first virtual router group and as a backup in the second virtual group. Then configure the second VRRP-E router to act as a backup in the first virtual group and as a master in the second virtual group.

This example is for VRRP-E. There are minor syntax differences for VRRP, which you can determine by consulting the appropriate command reference. The task steps below are configured on Router 1 and there are three configuration examples at the end of the task showing how to configure Router 1 as a backup and Router 2 as a master and a backup VRRP-E device.

1. Enter the **rbridge-id** command in global configuration mode, using the RBridge ID (which has an asterisk next to it when you run a **do show vcs** command).

```
device(config)# rbridge-id 101
```

2. Enable the VRRP-E protocol globally.

```
device(config-rbridge-id-101)# protocol vrrp-extended
```

3. Configure the virtual Ethernet (ve) interface link for Router 1.

```
device(config-rbridge-id-101)# interface ve 10
```

   NOTE
   You must first create **interface vlan 10** in global configuration mode.

4. Configure the IP address of the ve link for Router 1.

```
device(config-if-Ve-10)# ip address 192.168.4.2/24
```

5. To assign Router 1 to a VRRP-E group called Group 1, enter the command:

```
device(config-if-Ve-10)# vrrp-extended-group 1
```

6. Configure the tengigabitethernet port 101/2/4 as the tracking port for the interface ve 10, with a track priority of 20.

```
device(config-vrrp-extended-group-1)# track te 101/2/4 priority 20
```

7. Configure an IP address for the virtual router.

```
device(config-vrrp-extended-group-1)# virtual-ip 192.168.4.100
```

   NOTE
   (For VRRP-E only) The address you enter with the **virtual-ip** command cannot be the same as a real IP address configured on the interface.

8. To configure Router 1 as the master, set the priority to a value higher than the default (which is 100).

```
device(config-vrrp-group-1)# priority 110
```

Router 1 as backup

The following example configures Router 1 as a backup device for VRRP-E group 2 by configuring a priority (100) that is a lower value than the priority set for Router 2 in VRRP-E group 2.

```
device(config)# rbridge-id 101
device(config-rbridge-id-101)# protocol vrrp-extended
device(config-rbridge-id-101)# interface ve 10
device(config-Ve-10)# ip address 192.168.4.2/24
device(config-Ve-10)# vrrp-extended-group 2
device(config-vrrp-extended-group-1)# track te 101/2/4 priority 20
device(config-vrrp-extended-group-1)# virtual-ip 192.168.4.101
device(config-vrrp-group-1)# priority 100
```

Router 2 as master

The following example configures Router 2 as the master device for VRRP-E group 2 by configuring a priority (110) that is a higher value than the priority set for Router 1 in VRRP-E group 2.

```
device(config)# rbridge-id 102
device(config-rbridge-id-102)# protocol vrrp-extended
device(config-rbridge-id-102)# interface ve 10
device(config-Ve-10)# ip address 192.168.4.3/24
device(config-Ve-10)# vrrp-extended-group 2
device(config-vrrp-extended-group-2)# track te 101/2/4 priority 20
device(config-vrrp-extended-group-2)# virtual-ip 192.168.4.101
device(config-vrrp-group-1)# priority 110
```

Router 2 as backup

The following example configures Router 2 as a backup device for VRRP-E group 1 by configuring a priority (100) that is a lower value than the priority set for Router 1 in VRRP-E group 1.

```
device(config)# rbridge-id 102
device(config-rbridge-id-102)# protocol vrrp-extended
device(config-rbridge-id-102)# interface ve 10
device(config-Ve-10)# ip address 192.168.4.3/24
device(config-Ve-10)# vrrp-extended-group 1
device(config-vrrp-extended-group-1)# track te 101/2/4 priority 20
device(config-vrrp-extended-group-1)# virtual-ip 192.168.4.100
device(config-vrrp-group-1)# priority 100
```

Tracked ports and track priority with VRRP and VRRP-E

Port tracking allows interfaces not configured for VRRP or VRRP-E to be monitored for link-state changes that can result in dynamic changes to the VRRP device priority.

A tracked port allows you to monitor the state of the interfaces on the other end of a route path. A tracked interface also allows the virtual router to lower its priority if the exit path interface goes down, allowing another virtual router in the same VRRP (or VRRP-E) group to take over. When a tracked interface returns to an up state, the configured track priority is added to the current virtual router priority value. The following conditions and limitations exist for tracked ports:

- Track priorities must be lower than VRRP or VRRP-E priorities.
- The dynamic change of router priority can trigger a master device switchover if preemption is enabled. However, if the router is an owner, the master device switchover will not occur.
- The maximum number of interfaces that can be tracked for a virtual router is 16.

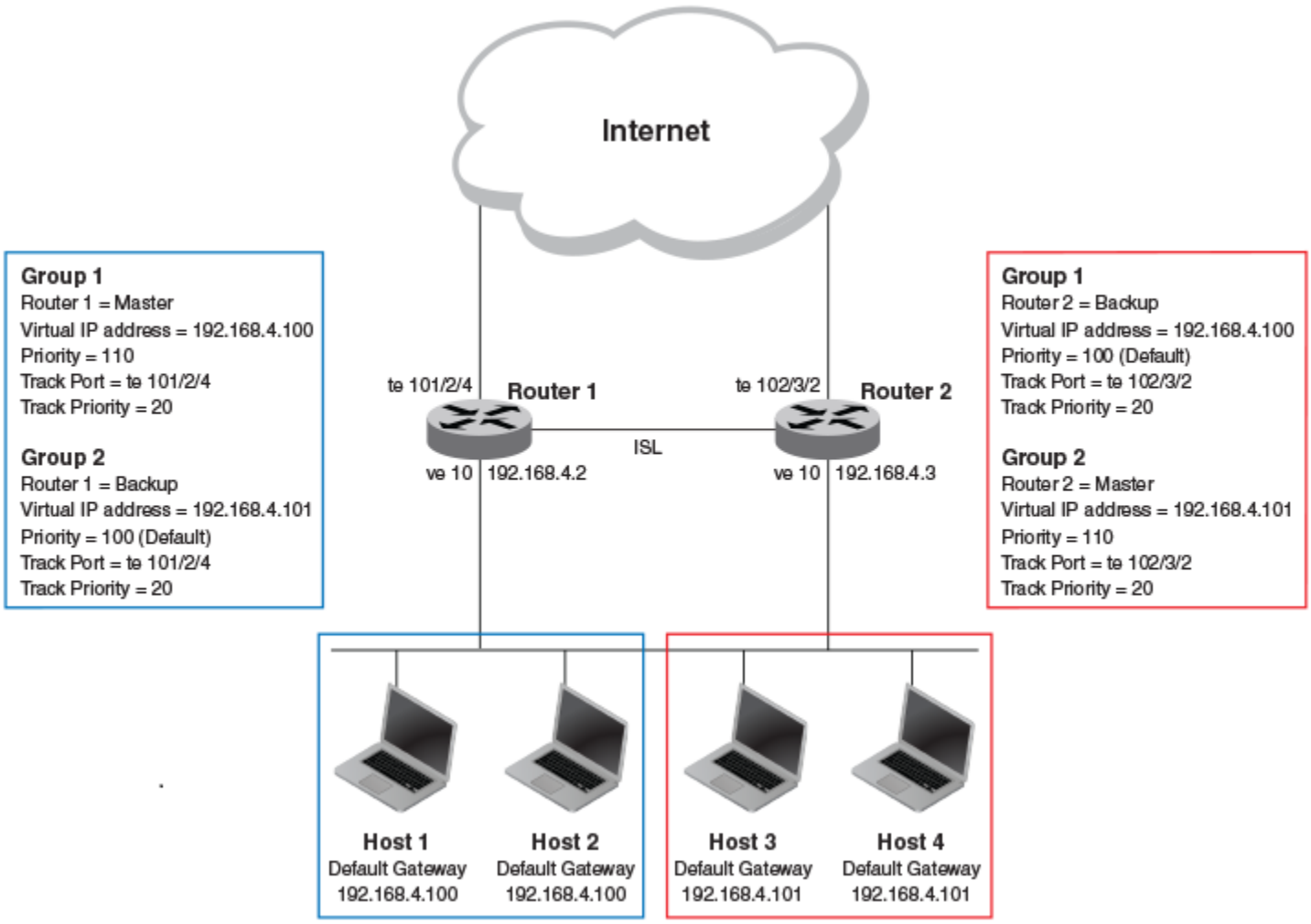
- Port tracking is allowed for physical interfaces and port channels.

Configuring VRRP port tracking

Configuring port tracking on an exit path interface and setting a priority on a VRRP device enables VRRP to monitor the interface. If the interface goes down, the device priority is lowered and another backup device with a higher priority assumes the role of master.

In the following task steps, interface tengigabitethernet 101/2/4 on Router 1 (shown in the diagram below) is configured to be tracked, and if the interface fails, the VRRP priority of Router 1 is lowered by a value of 20. Router 1 is the master VRRP device for group 1 and a lower priority triggers a backup device with a higher priority, Router 2, to become the new master for Group 1. Perform this task on the device on which the tracked interface exists.

FIGURE 55 Multigroup VRRP routing topology



1. Enter the **rbridge-id** command in global configuration mode, using the RBridge ID (which has an asterisk next to it when you run a **do show vcs** command).

```
device(config)# rbridge-id 101
```

2. Enable VRRP globally.

```
device(config-rbridge-id-101)# protocol vrrp
```

3. Enter interface configuration mode and run the following command:

```
device(config-rbridge-id-101)# interface ve 10
```

4. Run the following command to enter group configuration mode.

```
device(config-if-Ve-10)# vrrp-group 1
```

5. Enter the **track** command to set the track port and priority:

```
device(config-vrrp-group-1)# track te 101/2/4 priority 20
```

The following example shows how to configure an Ethernet interface on Router 2 to be tracked, and if the interface fails, the VRRP priority of Router 2 is lowered by a value of 40. Router 2 is the master VRRP device for group 2 and a lower priority triggers a backup device, Router 1, to become the new master for group 2.

```
device(config)# rbridge-id 102
device(config-rbridge-id-102)# protocol vrrp
device(config-rbridge-id-102)# interface ve 10
device(config-Ve-10)# vrrp-group 2
device(config-vrrp-group-1)# track te 101/2/4 priority 20
```

VRRP backup preemption

Preemption of a backup VRRP device acting as a master device is allowed when another backup device has a higher priority.

By default, preemption is enabled for VRRP. In VRRP, preemption allows a backup device with the highest priority to become the master device when the master (also the owner) device goes offline. If another backup device is added with a higher priority, it will assume the role of the master VRRP device. In some larger networks there may be a number of backup devices with varying levels of priority, and preemption can cause network flapping. To prevent the flapping, disable preemption.

> NOTE
> If preemption is disabled for VRRP, the owner device is not affected because the owner device always preempts the active master. When the owner device is online, the owner device assumes the role of the master device regardless of the setting for the preempt parameter.

In VRRP-E, preemption is disabled by default. In situations where a new backup device is to be added with a higher priority, preemption can be enabled. There are no owner devices in VRRP-E to automatically preempt a master device.

Enabling VRRP backup preemption

Allowing a backup VRRP device that is acting as the master to be preempted by another backup device with a higher priority value.

A VRRP session must be globally enabled using the **protocol vrrp** command in RBridge ID configuration mode.

Preemption is enabled by default for VRRP, and disabled by default on VRRP-E. Assuming that preemption is disabled in a VRRP session, perform the following steps on a VRRP backup device.

1. In RBridge ID configuration mode, configure the tengigabitethernet interface for the device.

```
device(config-rbridge-id-102)# interface tengigabitethernet 102/1/5
```

2. Configure the IP address of the interface:

```
device(conf-if-te-102/1/5)# ip address 192.168.4.3/24
```

NOTE
This router is a backup router.

3. Assign the device to VRRP group 1.

```
device(conf-if-te-102/1/5)# vrrp-group 1
```

4. Enter the **preempt-mode** command to configure backup preemption.

```
device(conf-vrrp-group-1)# preempt-mode
```

If a backup device has a higher priority than the current master device, the backup device will assume the role of the VRRP master device after preemption is enabled.

The following example enables preemption on a backup VRRP device.

```
device(config-rbridge-id-101)# interface tengigabitethernet 102/1/5
device(conf-if-te-101/1/5)# ip address 192.168.4.3/24
device(conf-if-te-101/1/5)# vrrp-group 1
device(config-vrrp-group-1)# preempt-mode
```

VRRP-Ev2 overview

VRRP Extended (VRRP-E) is an extended version of VRRP. VRRP-E is designed to avoid the limitations in the standards-based VRRP.

VRRP-E is implemented the following differences from RFC 3768 which describes VRRPv2 to provide extended functionality and ease of configuration:

- VRRP-E does not include the concept of an owner device, and a master VRRP-E is determined by the priority configured on the device.
- While the VRRP-E virtual router IP address must belong in the same subnet as a real IP address assigned to a physical interface of the device on which VRRP-E is configured, it must not be the same as any of the actual IP addresses on any interface.
- Configuring VRRP-E uses the same task steps for all devices; there are no differences between master and backup device configuration. The device configured with the highest priority assumes the master role.

NOTE
VRRP-E is supported on the devices described in this guide. In a mixed-device environment, consult your documentation for the other devices to determine if VRRP-E is supported.
VRRP-E does not interoperate with VRRP sessions.

Enabling a VRRP-E device

This task is performed on all devices that are designated as VRRP extended (VRRP-E) devices. While VRRP-E does not have owner devices, there is still a master device and backup devices with the master device determined by the device with the highest priority.

1. From privileged EXEC mode, enter configuration mode by issuing the **configure terminal** command.

```
device# configure terminal
```

2. Enter the **rbridge-id** command, using the RBridge ID (which has an asterisk next to it when you run a **do show vcs** command).

```
device(config)# rbridge-id 101
```

3. Globally enable VRRP-E.
   - Enter the **protocol vrrp** command on VDX 8770 devices.
   - Enter the **protocol vrrp-extended** command on VDX 6740 devices.

```
device(config-rbridge-id-101)# protocol vrrp
```

   or

```
device(config-rbridge-id-101)# protocol vrrp-extended
```

4. Configure the Virtual Ethernet (VE) interface link for the VRRP-E device.

```
device(config-rbridge-id-101)# interface ve 10
```

   Only ve interfaces are supported by VRRP-E.

5. Configure the IP address of the interface.

```
device(config-if-Ve-10)# ip address 192.168.4.1/24
```

6. Assign the device to a group called Group 1.

```
device(config-if-Ve-10)# vrrp-extended-group 1
```

   You can assign a group number in the range of 1 through 255.

7. Enter the **priority** command with a number to assign a priority.

```
device(config-vrrp-group-1)# priority 110
```

   The VRRP-E device with the highest priority number becomes the master device.

8. Assign a virtual router IP address.

```
device(config-vrrp-group-1)# virtual-ip 192.168.4.100
```

> NOTE
> For VRRP-E, the virtual router group IP address must not be the same as a real IP address configured on the interface.

Router 1

The following example configures a master VRRP-E device for group 1.

```
device# configure terminal
device(config)# rbridge-id 101
device(config-rbridge-id-101)# protocol vrrp-extended
device(config-rbridge-id-101)# interface ve 10
device(conf-ve-10)# ip address 192.168.4.1/24
device(conf-ve-10)# vrrp-extended-group 1
device(conf-ve-10)# priority 110
device(config-vrrp-group-1)# virtual-ip 192.168.4.100
```

Router 2

The following example configures a backup VRRP-E device for group 1. In the first configuration of VRRP-E for Router 1 the priority is set to 110, higher than the priority for Router 2 at 80. Router 1 assumes the role of the master VRRP-E device.

```
device# configure terminal
device(config)# rbridge-id 101
device(config-rbridge-id-101)# protocol vrrp-extended
device(config-rbridge-id-101)# interface ve 10
device(config-ve-10)# ip address 192.168.4.3/24
device(config-ve-10)# vrrp-extended-group 1
device(config-vrrp-group-1)# priority 80
device(config-vrrp-group-1)# virtual-ip 192.168.4.100
```

Track routes and track priority with VRRP-E

Route tracking allows networks not configured for VRRP extended (VRRP-E) to be monitored for network reachability changes that can result in dynamic changes to the VRRP-E device priority.

Using network addresses, routes are tracked for online or offline events. The networks to be tracked can be either present or absent from the Routing Information Base (RIB). When route-tracking is enabled in the configured VRRP-E instance, the status of the tracked route is monitored. The priority of the VRRP-E device may be changed dynamically due to the following events:

- When a tracked route goes into an offline state, the configured track priority is subtracted from the current value of the VRRP-E device.
- When a tracked route returns to an online state, the configured track priority is added to the current value of the VRRP-E device.

NOTE
Network tracking is not supported by VRRP; only VRRP-E supports network tracking.

The dynamic change of device priority can trigger a switchover from a master VRRP-E device to a backup VRRP-E device if preemption is enabled.

Forward referencing for tracked routes is supported. The tracked route can be removed and added without the need to reconfigure the tracking for the route.

NOTE
Maximum number of routes that can be tracked for a virtual VRRP-E device is 16.

Configuring VRRP-E route tracking

Configuring route tracking on an exit path network and setting a priority on a VRRP Extended (VRRP-E) device enables VRRP-E to monitor the route. If the network goes down, the device priority is lowered and another backup device with a higher priority assumes the role of master.

In the following task steps, network 10.1.1.0/24 is configured to be tracked, and if the network goes offline, the VRRP priority of the current master device is lowered by a value of 20.

1. In global configuration mode, enter the **rbridge-id** command using the RBridge ID (which has an asterisk next to it when you run a **do show vcs** command).

```
device(config)# rbridge-id 1
```

2. Enable VRRP-E globally.

```
device(config-rbridge-id-1)# protocol vrrp-extended
```

3. Enter interface configuration mode.

```
device(config-rbridge-id-1)# interface ve 100
```

4. Run the following command to enter group configuration mode.

```
device(config-if-Ve-100)# vrrp-extended-group 1
```

5. Enter the **track network** command to set the track network (route) and priority:

```
device(config-vrrp-group-1)# track network 10.1.1.0/24 priority 20
```

6. Return to privileged EXEC mode.

```
device(config-vrrp-group-1)# end
```

7. To view tracked networks with their priority and status, enter the following command:

```
device# show vrrp detail

============================================================================================
Rbridge-id:1
============================================================================================

Total number of VRRP session(s)   : 1

VRID 3
  Interface: Ve 100;  Ifindex: 1207959652
  Mode: VRRPE
.
.
.
  Hold time: 0 sec  (default: 0 sec)
  Master Down interval: 4 sec
  Trackport:
    Port(s)                    Priority  Port Status
    =======                    ========  ===========

  Tracknetwork:
    Network(s)                 Priority Status
    =======                    ======== ===========
    10.1.1.0/24                20       Down

  Global Statistics:
  ==================
    Checksum Error : 0
    Version Error  : 0
    VRID Invalid   : 0

  Session Statistics:
  ===================
    Advertisements             : Rx: 0, Tx: 0
    Neighbor Advertisements             : Tx: 0
.
.
.
```

The following example shows how to configure network 10.1.1.0/24 to be tracked. If the network goes down, the VRRP-E device priority is lowered by a value of 20. The lower priority may trigger a switchover and a backup device with a higher priority becomes the new master for VRRP-E group 1.

```
device(config)# rbridge-id 1
device(config-rbridge-id-1)# protocol vrrp-extended
device(config-rbridge-id-1)# interface ve 100
device(conf-Ve-100)# vrrp-extended-group 1
device(config-vrrp-group-1)# track network 10.1.1.0/24 priority 20
```

VRRP-E load-balancing using short-path forwarding

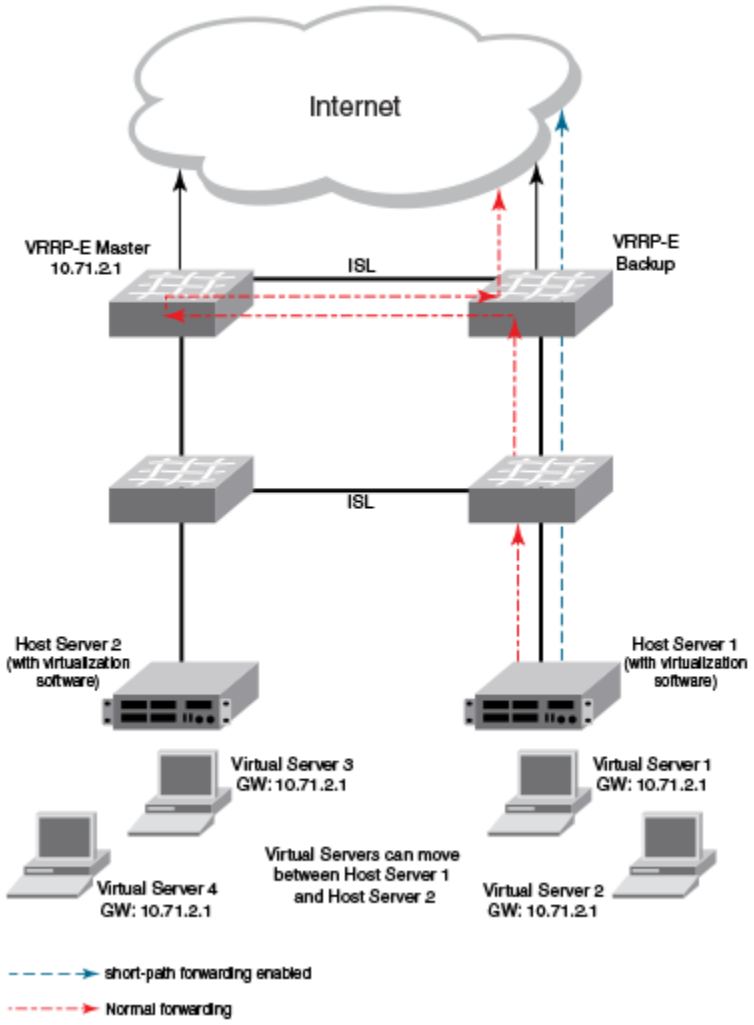
The VRRP-E Extension for Server Virtualization feature allows devices to bypass the VRRP-E master router and directly forward packets to their destination through interfaces on the VRRP-E backup router. This is called *short-path forwarding*. A backup router participates in a VRRP-E session only when short-path forwarding is enabled.

VRRP-E active-active load-balancing is available in VCS mode and uses flow-hashing techniques to determine the path. All nodes in the VCS are aware of all VRRP-E sessions and the participating RBridges in each session.

Packet routing with short-path forwarding to balance traffic load

When short-path forwarding is enabled, traffic load-balancing is performed because both master and backup devices can be used to forward packets.

FIGURE 56 Short-path forwarding



If you enable short-path forwarding in both master and backup VRRP-E devices, packets sent by Host Server 1 (in the figure) and destined for the Internet cloud through the device on which a VRRP-E backup interface exists can be routed directly to the VRRP-E backup device (blue dotted line) instead of being switched to the master router and then back (red dotted-dash line).

In the figure, load-balancing is achieved using short-path forwarding by dynamically moving the virtual servers between Host Server 1 and Host Server 2.

Short-path forwarding with revert priority

Revert priority is used to dynamically enable or disable VRRP-E short-path forwarding.

If short-path forwarding is configured with revert priority on a backup router, the revert priority represents a threshold for the current priority of the VRRP-E session. When the backup device priority is higher than the configured revert priority, the backup router is able to perform short-path forwarding. If the backup priority is lower than the revert priority, short-path forwarding is disabled.

Configuring VRRP-E load-balancing using short-path forwarding

VRRP-E traffic can be load-balanced using short-path forwarding on the backup devices.

Before configuring VRRP-E load-balancing, VRRP-E must be configured on all devices in the VRRP-E session.

Perform this task on all backup VRRP-E Layer 3 devices to allow load sharing within a VRRP extended group.

1. From privileged EXEC mode, enter global configuration mode.

   ```
   device# configure terminal
   ```

2. Enter the **rbridge-id** command with an associated RBridge ID to enter RBridge configuration mode for a specific ID.

   ```
   device(config)# rbridge-id 122
   ```

3. Globally enable VRRP-E.

   ```
   device(config-rbridge-id-122)# protocol vrrp-extended
   ```

4. Enter the **interface ve** command with an associated VLAN number.

   ```
   device(config-rbridge-id-122)# interface ve 2019
   ```

   In this example, virtual Ethernet (ve) configuration mode is entered and the interface is assigned with a VLAN number of 2019.

5. Enter an IP address for the interface using the **ip address** command.

   ```
   device(config-ve-2019)# ip address 192.168.4.1/24
   ```

6. Enter the **vrrp-extended-group** command with a number to assign a VRRP-E group to the device.

   ```
   device(config-ve-2018)# vrrp-extended-group 19
   ```

   In this example, VRRP-E group configuration mode is entered.

7. Enter the **short-path-forwarding** command with a **revert-priority** value to configure the backup VRRP-E as an alternate path with a specified priority.

   ```
   device(config-vrrp-extended-group-19)# short-path-forwarding revert-priority 50
   ```

   When the backup device priority is higher than the configured **revert-priority** value, the backup router is able to perform short-path forwarding. If the backup priority is lower than the revert priority, short-path forwarding is disabled.

In the following example, short-path forwarding is configured on a backup VRRP-E device and a revert priority threshold is configured. If the backup device priority falls below this threshold, short-path forwarding is disabled.

```
device# configure
device(config)# rbridge-id 122
device(config-rbridge-id-122)# protocol vrrp-extended
device(config-rbridge-id-122)# interface ve 2019
device(config-ve-2019)# ip address 192.168.4.1/24
device(config-ve-2019)# vrrp-extended-group 19
device(config-vrrp-extended-group-19)# short-path-forwarding revert-priority 50
```

Displaying VRRPv2 information

Various show commands can be used to display statistical and summary information about VRRP and VRRP-E configurations.

Before displaying VRRP information, VRRPv2 must be configured and enabled in your VRRP or VRRP-E network to generate traffic.

Use one or more of the following commands to display VRRPv2 information. The commands do not have to be entered in this order.

1. Enter the **show vrrp** command with a virtual-group ID to display detailed information about one virtual group ID.

```
device# show vrrp 1

============Rbridge-id:2============

Total number of VRRP session(s)   : 1

VRID 1
  Interface: Ve 10;  Ifindex: 1207959562
  Mode: VRRP
  Admin Status: Enabled
  Description :
  Address family: IPv4
  Version: 2
  Authentication type: No Authentication
  State: Initialize
  Session Master IP Address:
  Virtual IP(s): 192.168.4.1
  Configured Priority: unset (default: 100); Current Priority: 100
  Advertisement interval: 1 sec  (default: 1 sec)
  Preempt mode: ENABLE  (default: ENABLE)
  Hold time: 0 sec  (default: 0 sec)
  Trackport:
    Port(s)                   Priority  Port Status
    =======                   ========  ===========
  Statistics:
    Advertisements: Rx: 60, Tx: 6
    Gratuitous ARP: Tx: 2
```

   This example output shows that one IPv4 VRRP session is configured.

2. Enter the **show vrrp summary** command.

```
device# show vrrp summary

============Rbridge-id:2============

Total number of VRRP session(s)   : 1
Master session count  : 1
Backup session count  : 0
Init session count    : 0

VRID  Session  Interface       Admin     Current   State    Short-path  Revert    SPF
                               State     Priority            Forwarding  Priority  Reverted
====  =======  =========       =====     ========  ======   ==========  ========  ========
1     VRRP     Ve 100          Enabled   110       Master
```

   This example displays information about VRRP sessions.

3. Enter the **show vrrp interface** command with interface ve 10 options and detailed output.

```
device# show vrrp int ve 10 detail

============Rbridge-id:2============

Total number of VRRP session(s)   : 1

VRID 1
  Interface: Ve 10;  Ifindex: 1207959562
  Mode: VRRP
  Admin Status: Enabled
  Description :
  Address family: IPv4
  Version: 2
  Authentication type: No Authentication
  State: Initialize
  Session Master IP Address:
  Virtual IP(s): 192.168.4.1
  Virtual MAC Address: 0000.5e00.0101
  Configured Priority: 110 (default: 100); Current Priority: unset
  Advertisement interval: 1 sec  (default: 1 sec)
  Preempt mode: ENABLE  (default: ENABLE)
  Hold time: 0 sec  (default: 0 sec)
  Master Down interval: 4 sec
  Trackport:
    Port(s)                   Priority  Port Status
    =======                   ========  ===========

  Global Statistics:
  ==================
    Checksum Error : 0
    Version Error  : 0
    VRID Invalid   : 0

  Session Statistics:
  ===================
    Advertisements             : Rx: 60, Tx: 6
    Gratuitous ARP             : Tx: 2
    Session becoming master    : 0
    Advts with wrong interval  : 0
    Prio Zero pkts             : Rx: 0, Tx: 0
    Invalid Pkts Rvcd          : 0
    Bad Virtual-IP Pkts        : 0
    Invalid Authenticaton type : 0
    Invalid TTL Value          : 0
    Invalid Packet Length      : 0
```

Clearing VRRPv2 statistics

VRRPv2 session counters can be cleared using a CLI command.

Ensure that VRRPv2 or VRRP-Ev2 is configured and enabled in your network.

To determine the effect of clearing the VRRP statistics, an appropriate **show** command is entered before and after the **clear** command.

1. Enter the **end** or **exit** command to return to privileged EXEC mode.

2. Enter the **show vrrp** command with a virtual-group ID.

```
device# show vrrp 1

============Rbridge-id:125============

Total number of VRRP session(s)   : 2

VRID 1
  Interface: Ve 10;  Ifindex: 1207959562
  Mode: VRRP
  Admin Status: Enabled
  Description :
  Address family: IPv4
  Version: 2
.
.
.
Statistics:
    Advertisements: Rx: 0, Tx: 60
    Neighbor Advertisements: Tx: 30
```

3. Enter the **clear vrrp statistics** command.

```
device# clear vrrp statistics
```

4. Enter the **show vrrp** command with a virtual-group ID.

```
device# show vrrp 1

============Rbridge-id:125============

Total number of VRRP session(s)   : 2

VRID 1
  Interface: Ve 10;  Ifindex: 1207959562
  Mode: VRRP
  Admin Status: Enabled
  Description :
  Address family: IPv4
  Version: 2
.
.
.
Statistics:
    Advertisements: Rx: 0, Tx: 6
    Neighbor Advertisements: Tx: 3
```

In this show output after the **clear vrrp statistics** command has been entered, you can see that the statistical counters have been reset. Although some of the counters are showing numbers because VRRP traffic is still flowing, the numbers are much lower (6 transmissions instead of 60 transmissions) than in the initial **show vrrp** command output.

VRRPv3

VRRPv3 overview

VRRP version 3 (VRRPv3) introduces IPv6 address support for both standard VRRP and VRRP enhanced (VRRP-E).

Virtual Router Redundancy Protocol (VRRP) is designed to eliminate the single point of failure inherent in a static default routed environment by providing redundancy to Layer 3 devices within a local area network (LAN). VRRP uses an election protocol to dynamically assign the default gateway for a host to one of a group of VRRP routers on a LAN. Alternate gateway router paths can be allocated without changing the IP address or MAC address by which the host device knows its gateway.

VRRPv3 implements support for IPv6 addresses for networks using IPv6, and it also supports IPv4 addresses for dual-stack networks configured with VRRP or VRRP-E. VRRPv3 is compliant with RFC 5798. The benefit of implementing VRRPv3 is faster switchover to backup devices than can be achieved using standard IPv6 neighbor discovery mechanisms. With VRRPv3, a backup router can become a master router in a few seconds with less overhead traffic and no interaction with the hosts.

When VRRPv3 is configured, the master device that owns the virtual IP address and a master device that does not own the virtual IP address can both respond to ICMP echo requests (using the **ping** command) and accept Telnet and other management traffic sent to the virtual IP address. In VRRPv2, only a master device on which the virtual IP address is the address of an interface on the master device can respond to ping and other management traffic.

The following are other IPv6 VRRPv3 functionality details:

- VRRPv2 functionality is supported by VRRPv3 except for VRRP authentication.
- Two VRRP and VRRP-E sessions cannot share the same group ID on the same interface.

NOTE

When implementing IPv6 VRRPv3 across a network with devices from other vendors, be aware of a potential interoperability issue with IPv6 VRRPv3 and other vendor equipment. Extreme has implemented IPv6 VRRPv3 functionality to comply with RFC 5798 and will interoperate comfortably with other vendors that support RFC 5798.

VRRPv3 functionality differences on VDX devices

The implementation of VRRPv3 varies across the VDX products.

VRRPv3 functionality on the VDX 8770 platforms.

- VRRP and VRRP-E configurations can coexist on the same interface.

VRRPv3 functionality on the VDX 6740 platforms.

- VRRP and VRRP-E configurations cannot coexist on the same interface.

VRRPv3 functionality on the VDX 6940 platforms.

- IPv4 VRRP or IPv6 VRRP can be individually configured along with VRRP-E IPv4 and IPv6, but not IPv4 VRRP and IPv6 VRRP simultaneously.
- If IPv4 VRRP and IPv6 VRRP are both configured, then VRRP-E cannot be configured.

VRRPv3 performance and scalability metrics for Network OS devices

The following table defines VRRPv3 system resource metrics by Network OS device.

TABLE 20 System resource metrics for VRRPv3

System resource	VDX 87xx	VDX 67xx	VDX 6940-36Q, 6940-144S
Max # of VRRP and VRRP-E sessions with advertisement interval of 1 second	1024	255	512
Max # of VRRP and VRRP-E sessions with advertisement interval of 500 milliseconds	500	100	200
Max # of VRRP and VRRP-E sessions with advertisement interval of 200 milliseconds	200	50	100
VRRP sessions per interface	32	32	16
Max # of VRRP devices	8	8	8

Enabling IPv6 VRRPv3

IPv6 VRRPv3 is enabled on a device when a virtual IPv6 address is assigned to a VRRPv3 group.

Before assigning a virtual IPv6 address to a VRRPv3 group, you must configure IPv6 VRRP version 3 on a virtual Ethernet interface and assign a VRRPv3 group to the device. The VRRPv3 session is enabled using a virtual IPv6 address. The device must be a router or another device that supports Layer 3 routing.

Perform this task on all devices that are to run VRRPv3. The device to which the virtual IP address belongs determines the initial master device status with all the other devices acting as backups.

1. Enter the **configure** command to access global configuration mode.

```
device# configure
```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

```
device(config)# rbridge-id 125
```

3. To globally enable VRRPv3, enter the **ipv6 protocol vrrp** command.

```
device(config-rbridge-id-125)# ipv6 protocol vrrp
```

4. Enter the **interface ve** command with an associated VLAN number.

```
device(config-rbridge-id-125)# interface ve 2018
```

   In this example, virtual Ethernet (ve) interface configuration mode is entered and the interface is assigned with a VLAN number of 2018.

5. Enter an IPv6 address for the interface using the **ipv6 address** command.

```
device(config-ve-2018)# ipv6 address 2001:2018:8192::125/64
```

6. Enter the **ipv6 vrrp-group** command with a number to assign a VRRPv3 group to the device.

```
device(config-ve-2018)# ipv6 vrrp-group 18
```

   In this example, VRRP group configuration mode is entered.

7. Enter the **virtual-ip** command to assign a link-local virtual IPv6 address to a VRRPv3 group.

```
device(config-vrrp-group-18)# virtual-ip fe80::2018:1
```

   In this example, the link-local IPv6 address of the virtual router is assigned to VRRPv3 group 18. The first virtual IP address entered enables the VRRPv3 session.

   NOTE
   A link-local IPv6 address is valid only for a single network link. If the virtual IP address can be reached from outside the local network, a global IPv6 address must be configured as a virtual IP address. At least one link-local address is also required.

8. Enter the **virtual-ip** command to assign a virtual IPv6 address to a VRRPv3 group.

```
device(config-vrrp-group-18)# virtual-ip 2001:2018:8192::1
```

   In this example, the IPv6 address of the virtual router is assigned to VRRPv3 group 18.

The following example shows how to enable a VRRPv3 session by assigning virtual IP addresses to a VRRPv3 virtual group.

```
device# configure
device(config)# rbridge-id 125
device(config-rbridge-id-125)# ipv6 protocol vrrp
device(config-rbridge-id-125)# interface ve 2018
device(config-ve-2018)# ipv6 address 2001:2018:8192::122/64
device(config-ve-2018)# ipv6 vrrp-group 18
device(config-vrrp-group-18)# virtual-ip fe80::2018:1
device(config-vrrp-group-18)# virtual-ip 2001:2018:8192::1
```

Enabling IPv4 VRRPv3

IPv4 VRRPv3 is enabled on a device when a virtual IP address is assigned to a VRRPv3 group.

VRRPv3 supports IPv4 sessions as well as IPv6 sessions. To configure a VRRPv3 session for IPv4 assign a virtual router group with the **v3** option to the device. The device must be a router or another device that supports Layer 3 routing.

Perform this task on all devices that are to run IPv4 VRRPv3. The device to which the virtual IP address belongs determines the initial master device status with all the other devices acting as backups.

1. Enter the **configure terminal** command to access global configuration mode.

```
device# configure terminal
```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

```
device(config)# rbridge-id 101
```

3. To globally enable VRRP, enter the **protocol vrrp** command.

```
device(config-rbridge-id-101)# protocol vrrp
```

4. Configure the Ethernet interface.

```
device(config-rbridge-id-101)# interface tengigabitethernet 101/1/6
```

   In this example, tengigabitethernet (te) interface configuration mode is entered and the interface is assigned with an associated RBridge ID and slot/port number of 101/1/6.

5. Enter an IPv4 address for the interface using the **ip address** command.

```
device(config-if-te-101/1/6)# ip address 192.168.5.2/24
```

6. Enter the **vrrp-group** command with a number to assign a virtual router group to the device and a version to configure VRRPv3.

```
device(config-if-te-101/1/6)# vrrp-group 10 version 3
```

   In this example, a VRRPv3 group is assigned and VRRP group configuration mode is entered.

7. Enter the **advertisement-interval** command with a number in milliseconds to configure the interval at which the master VRRP router advertises its existence to the backup routers.

```
device(config-vrrp-group-10)# advertisement-interval 2000
```

   In this example, the interval is expressed as 2000 milliseconds because VRRPv3 uses milliseconds instead of seconds for the advertisement interval.

8. Enter the **virtual-ip** command to assign a virtual IP address to a VRRPv3 group.

```
device(config-vrrp-group-10)# virtual-ip 192.168.5.2
```

   In this example, the IPv4 address of the virtual router is assigned to VRRPv3 group 10. This virtual IP address belongs to this device and this device will assume the role of the master device.

The following example shows how to enable an IPv4 VRRPv3 session by assigning virtual IP addresses to a VRRPv3 virtual group.

```
device# configure
device(config)# rbridge-id 101
device(config-rbridge-id-101)# protocol vrrp
device(config-rbridge-id-101)# interface tengigabitethernet 101/1/6
device(config-if-te-101/1/6)# ip address 192.168.5.2/24
device(config-if-te-101/1/6)# vrrp-group 10 version 3
device(config-vrrp-group-10)# advertisement-interval 2000
device(config-vrrp-group-10)# virtual-ip 192.168.5.2
```

Tracked ports and track priority with VRRP and VRRP-E

Port tracking allows interfaces not configured for VRRP or VRRP-E to be monitored for link-state changes that can result in dynamic changes to the VRRP device priority.

A tracked port allows you to monitor the state of the interfaces on the other end of a route path. A tracked interface also allows the virtual router to lower its priority if the exit path interface goes down, allowing another virtual router in the same VRRP (or VRRP-E) group to take over. When a tracked interface returns to an up state, the configured track priority is added to the current virtual router priority value. The following conditions and limitations exist for tracked ports:

- Track priorities must be lower than VRRP or VRRP-E priorities.
- The dynamic change of router priority can trigger a master device switchover if preemption is enabled. However, if the router is an owner, the master device switchover will not occur.
- The maximum number of interfaces that can be tracked for a virtual router is 16.
- Port tracking is allowed for physical interfaces and port channels.

Port tracking using IPv6 VRRPv3

The tracking of the link status of an interface not configured for VRRP or VRRP-E can be configured with a priority that can result in dynamic changes to the VRRP device priority.

After enabling IPv6 VRRPv3 you can configure tracking the port status of other interfaces on the device that are not configured for VRRP. Any link down or up events from tracked interfaces can result in dynamic changes in the virtual router priority and a potential master device switchover. The configured priority must be less than the VRRPv3 or VRRP-Ev3 priorities.

1. Enter the **configure** command to access global configuration mode.

   ```
   device# configure
   ```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

   ```
   device(config)# rbridge-id 125
   ```

3. To globally enable VRRPv3, enter the **ipv6 protocol vrrp** command.

   ```
   device(config-rbridge-id-125)# ipv6 protocol vrrp
   ```

4. Enter the **interface ve** command with an associated VLAN number.

   ```
   device(config-rbridge-id-125)# interface ve 2018
   ```

   In this example, virtual Ethernet (ve) interface configuration mode is entered and the interface is assigned with a VLAN number of 2018.

5. Enter an IPv6 address for the interface using the **ipv6 address** command.

   ```
   device(config-ve-2018)# ipv6 address 2001:2018:8192::125/64
   ```

6. Enter the **ipv6 vrrp-group** command with a number to assign a VRRPv3 group to the device.

   ```
   device(config-ve-2018)# ipv6 vrrp-group 18
   ```

   In this example, VRRP group configuration mode is entered.

7. Enter the **virtual-ip** command to assign a link-local virtual IPv6 address to a VRRPv3 group.

```
device(config-vrrp-group-18)# virtual-ip fe80::2018:1
```

   In this example, the link-local IPv6 address of the virtual router is assigned to VRRPv3 group 18. The first virtual IP address entered enables the VRRPv3 session.

8. Enter the **virtual-ip** command to assign a virtual IPv6 address to a VRRPv3 group.

```
device(config-vrrp-group-18)# virtual-ip 2001:2018:8192::1
```

   In this example, the IPv6 address of the virtual router is assigned to VRRPv3 group 18.

9. Enter the **track** command with an interface and a priority to enable the tracking of ports that are not configured as VRRP interfaces.

```
device(config-vrrp-group-18)# track tengigabitethernet 3/0/5 priority 15
```

10. Enter the **no preempt-mode** command to disable preemption.

```
device(config-vrrp-group-18)# no preempt-mode
```

    Preemption can be disabled when you do not want to preempt an existing master with a higher priority device.

11. Enter the **priority** command to configure the priority of the virtual router. In VRRPv3, the virtual router with the highest priority becomes the master VRRPv3 device.

```
device(config-vrrp-group-18)# priority 120
```

The following example shows how to configure an IPv6 VRRPv3 session and enable the tracking of a 10 GbE interface.

```
device# configure
device(config)# rbridge-id 125
device(config-rbridge-id-125)# ipv6 protocol vrrp
device(config-rbridge-id-125)# interface ve 2018
device(config-ve-2018)# ipv6 address 2001:2018:8192::122/64
device(config-ve-2018)# ipv6 vrrp-group 18
device(config-vrrp-group-18)# virtual-ip fe80::2018:1
device(config-vrrp-group-18)# virtual-ip 2001:2018:8192::1
device(config-vrrp-group-18)# track tengigabitethernet 3/0/5 priority 15
device(config-vrrp-group-18)# no preempt-mode
device(config-vrrp-group-18)# priority 120
```

VRRP hold timer

The hold timer delays the preemption of a master VRRP device by a high-priority backup device.

A hold timer is used when a VRRP-enabled device that was previously a master device failed, but is now back up. This restored device now has a higher priority than the current VRRP master device, and VRRP normally triggers an immediate switchover. In this situation, it is possible that not all software components on the backup device have converged yet. The hold timer can enforce a waiting period before the higher-priority backup device assumes the role of master VRRP device again. The timer must be set to a number greater than 0 seconds for this functionality to take effect.

Hold timer functionality is supported in both version 2 and version 3 of VRRP and VRRP-E.

Configuring VRRP hold timer support

A hold timer can be configured on a VRRP-enabled interface to set an interval, in seconds, before a backup device becomes the master VRRP device.

A hold timer is used when a VRRP-enabled device that was previously a master device failed, but is now back online. The backup device has a higher priority than the current VRRP master device. Before assuming the role of master VRRP device again, the backup device waits for the time period specified in the hold timer. This task is supported in both versions of VRRP and VRRP-E, but the configuration below is for VRRPv3.

1. Enter the **configure terminal** command to access global configuration mode.

   ```
   device# configure terminal
   ```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

   ```
   device(config)# rbridge-id 125
   ```

3. Enable IPv6 VRRP-E.

   ```
   device(config-rbridge-id-125)# ipv6 protocol vrrp-extended
   ```

   In this example, virtual Ethernet (ve) interface configuration mode is entered and the interface is assigned with a VLAN number of 2018.

4. Enter the **interface ve** command with an associated vlan number.

   ```
   device(config-rbridge-id-125)# interface ve 2018
   ```

   In this example, virtual Ethernet (ve) interface configuration mode is entered and the interface is assigned with a VLAN number of 2018.

5. Enter an IPv6 address for the interface using the **ipv6 address** command.

   ```
   device(config-ve-2018)# ipv6 address 2001:2018:8192::122/64
   ```

6. Enter the **ipv6 vrrp-group** command with a number to assign a VRRPv3 group to the device.

   ```
   device(config-ve-2018)# ipv6 vrrp-group 18
   ```

   In this example, VRRP group configuration mode is entered.

7. Enter the **description** command to enter text that describes the virtual router group.

   ```
   device(config-vrrp-group-18)# description Product Marketing group
   ```

8. Enter the **advertisement-interval** command with a number representing milliseconds.

   ```
   device(config-vrrp-group-18)# advertisement-interval 3000
   ```

   NOTE
   In VRRPv3, the advertisement-interval is in milliseconds.

9. Enter the **hold-time** command with a number representing seconds.

   ```
   device(config-vrrp-group-18)# hold-time 5
   ```

The following example configures and enables a VRRPv3 session and adds a VRRP group description. A hold time of 5 seconds is configured. This example also contains appropriate **virtual-ip** command configuration not included in the task above.

```
device# configure
device(config)# rbridge-id 125
device(config-rbridge-id-125)# ipv6 protocol vrrp-extended
device(config-rbridge-id-125)# interface ve 2018
device(config-ve-2018)# ipv6 address 2001:2018:8192::122/64
device(config-ve-2018)# ipv6 vrrp-group 18
device(config-vrrp-group-18)# virtual-ip fe80::2018:1
device(config-vrrp-group-18)# virtual-ip 2001:2018:8192::1
device(config-vrrp-group-18)# description Product Marketing group
device(config-vrrp-group-18)# advertisement-interval 3000
device(config-vrrp-group-18)# hold-time 5
```

Alternate VRRPv2 checksum for VRRPv3 IPv4 sessions

If VRRPv3 is configured on an Extreme device in a network with third-party peering devices using VRRPv2-style checksum calculations for IPv4 VRRPv3 sessions, a VRRPv2-style checksum must be configured for VRRPv3 IPv4 sessions on the device.

VRRPv3 introduced a new checksum method for both IPv4 and IPv6 sessions, and this version 3 checksum computation is enabled by default. To accommodate third-party devices that still use a VRRPv2-style checksum for IPv4 VRRPv3 sessions, a command-line interface (CLI) command is available for configuration on a device. The new version 2 checksum method is disabled by default and is applicable only to IPv4 VRRPv3 sessions. If configured for VRRPv2 sessions, the VRRPv2-style checksum command is accepted, but it has no effect.

Enabling the v2 checksum computation method in a VRRPv3 IPv4 session

Enabling the alternate VRRPv2-style checksum in a VRRPv3 IPv4 session for compatibility with third-party network devices.

VRRPv3 uses the v3 checksum computation method by default for both IPv4 and IPv6 sessions on this device. Third-party devices may only have a VRRPv2-style checksum computation available for a VRRPv3 IPv4 session. The **use-v2-checksum** command is entered in interface configuration mode.

1. Enter the **configure terminal** command to enter global configuration mode.

   ```
   device# configure terminal
   ```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

   ```
   device(config)# rbridge-id 125
   ```

3. To enable VRRP globally enter the **protocol vrrp** command.

   ```
   device(config-rbridge-id-125)# protocol vrrp
   ```

4. Enter the **interface ve** command with an associated VLAN number.

   ```
   device(config-rbridge-id-125)# interface ve 2018
   ```

5. To assign an IPv4 VRRPv3 group to the device use the **vrrp-group** command with a group number and version 3.

   ```
   device(config-ve-2018)# vrrp-group 10 version 3
   ```

6. To enable v2 checksum computation method in an IPv4 VRRPv3 session, use the **use-v2-checksum** command in the VRRP group configuration mode.

```
device(config-vrrp-group-10)# use-v2-checksum
```

The following example shows the v2 checksum computation method enabled for an VRRPv3 IPv4 session on a device.

```
device# configure terminal
device(config)# rbridge-id 125
device(config-rbridge-id-125)# protocol vrrp
device(config-rbridge-id-125)# interface ve 2018
device(config-ve-2018)# vrrp-group 10 version 3
device(config-vrrp-group-10)# use-v2-checksum
```

VRRPv3 router advertisement suppression

VRRPv3 introduces the ability to suppress router advertisements (RAs).

Router advertisements are sent by the VRRP master device and contain the link-local virtual IP address and the virtual MAC address. For network security reasons, if you do not want the MAC addresses of interfaces to be viewed, you can disable RA messages. Disabling RA does not remove the auto-configured addresses being sent by VRRP updates, but the RA messages are dropped by the router interface. There are two other situations where you may want to disable RA messages:

- If an interface is currently the VRRP master but the virtual IP address is not the address of this interface, the device should not send RA messages for the interface IP address.
- If the interface is in a backup state, the device should not send RA messages for the interface IP address.

Disabling VRRPv3 router advertisements

The ability to suppress VRRPv3 master device interface router advertisements is introduced.

Suppressing interface router advertisements from the master VRRPv3 device may be performed for network security concerns because the RA messages include the MAC addresses of interfaces. In this task, VRRP-Ev3 is configured globally and RA messages are suppressed for the virtual ethernet (VE) 2109 interface.

NOTE
To configure this task for VRRPv3, use the **ipv6 protocol vrrp** command.

1. Enter the **configure terminal** command to access global configuration mode.

```
device# configure terminal
```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

```
device(config)# rbridge-id 122
```

3. Globally enable VRRP-Ev3.

```
device(config-rbridge-id-122)# ipv6 protocol vrrp-extended
```

4. Enter the **interface ve** command with an associated VLAN number.

```
device(config-rbridge-id-122)# interface ve 2019
```

In this example, virtual Ethernet (ve) configuration mode is entered and the interface is assigned with a VLAN number of 2019.

5. Enter the **ipv6 vrrp-suppress-interface-ra** command to suppress interface RA messages for the ve 2019 interface.

```
device(config-ve-2019)# ipv6 vrrp-suppress-interface-ra
```

The following example shows how to disable VRRPv3 RA messages from interface configuration mode for a VRRP-Ev3 session.

```
device# configure
device(config)# rbridge-id 122
device(config-rbridge-id-122)# ipv6 protocol vrrp-extended
device(config-rbridge-id-122)# interface ve 2019
device(config-ve-2019)# ipv6 vrrp-suppress-interface-ra
```

Displaying VRRPv3 statistics

Various show commands can display statistical information about IPv6 VRRP configurations.

Before displaying statistics, VRRPv3 must be configured and enabled in your network to generate traffic.

Use one or more of the following commands to display VRRPv3 information. The commands do not have to be entered in this order.

1. Use the **exit** command to return to privileged EXEC mode, if required.
2. Enter the **show ipv6 vrrp summary** command.

```
device# show ipv6 vrrp summary

============Rbridge-id:122============

Total number of VRRP session(s)    : 2
Master session count  : 1
Backup session count  : 1
Init session count    : 0

VRID  Session  Interface  Admin    Current   State   Short-path  Revert    SPF
                          State    Priority          Forwarding  Priority  Reverted
====  =======  =========  =====    ========  =====   ==========  ========  ========
18    VRRPE    Ve 2018    Enabled  254       Master  Enabled     unset     No
19    VRRPE    Ve 2019    Enabled  100       Backup  Enabled     unset     No
```

This example shows summary output for the two IPv6 VRRP-E sessions that are configured for virtual routers 18 and 19.

3. To display detailed information for a single VRRP virtual router ID(VRID), enter the **show ipv6 vrrp** command with the **detail** keyword and a specific VRID.

```
device# show ipv6 vrrp 19 detail

============Rbridge-id:122============

Total number of VRRP session(s)   : 1

VRID 19
  Interface: Ve 2019;  Ifindex: 1207961571
  Mode: VRRPE
  Admin Status: Enabled
  Description :
  Address family: IPv6
  Version: 3
  Authentication type: No Authentication
  State: Backup
  Session Master IP Address: fe80::205:33ff:fe79:fb1e
  Virtual IP(s): 2001:2019:8192::1
  Virtual MAC Address: 02e0.5200.2513
  Configured Priority: unset (default: 100); Current Priority: 100
  Advertisement interval: 1 sec  (default: 1 sec)
  Preempt mode: DISABLE  (default: DISABLED)
  Advertise-backup: ENABLE  (default: DISABLED)
  Backup Advertisement interval: 60 sec  (default: 60 sec)
  Short-path-forwarding: Enabled
  Revert-Priority: unset; SPF Reverted: No
  Hold time: 0 sec  (default: 0 sec)
  Master Down interval: 4 sec
  Trackport:
    Port(s)                   Priority  Port Status
    =======                   ========  ===========

  Global Statistics:
  ==================
    Checksum Error : 0
    Version Error  : 0
    VRID Invalid   : 0

  Session Statistics:
  ===================
    Advertisements             : Rx: 103259, Tx: 1721
    Neighbor Advertisements             : Tx: 0
    Session becoming master    : 0
    Advts with wrong interval  : 0
    Prio Zero pkts             : Rx: 0, Tx: 0
    Invalid Pkts Rvcd          : 0
    Bad Virtual-IP Pkts        : 0
    Invalid Authenticaton type : 0
    Invalid TTL Value          : 0
    Invalid Packet Length      : 0
    VRRPE backup advt sent     : 1721
    VRRPE backup advt recvd    : 0
```

This example shows detailed output for the IPv6 VRRP-E session for virtual router 19.

Clearing VRRPv3 statistics

VRRPv3 session counters can be cleared using a CLI command.

Ensure that VRRPv3 is configured and enabled in your network.

An excerpt of the appropriate VRRP show command output is shown before and after the appropriate VRRP clear command is entered to demonstrate that counters are cleared.

1. Enter the **end** command, if required, to return to privileged EXEC mode.
2. Enter the **show ipv6 vrrp detail** as in the following example.

```
device# show ipv6 vrrp detail

================================================================================
Rbridge-id:14
================================================================================

Total number of VRRP session(s)   : 1

VRID 11
  Interface: Ve 100;  Ifindex: 1207959652
  Mode: VRRP
  Admin Status: Enabled
  Description :
  Address family: IPv6
  Version: 3
  Authentication type: No Authentication
  State: Master
  Session Master IP Address: Local
  Backup Router(s): fe80::205:33ff:fe65:9d44
  Virtual IP(s): fe80::1
  Virtual MAC Address: 02e0.5200.250b
  Configured Priority: unset (default: 100); Current Priority: 100
  Advertisement interval: 1 sec  (default: 1 sec)
  Preempt mode: ENABLE  (default: DISABLED)
  Advertise-backup: ENABLE  (default: DISABLED)
  Backup Advertisement interval: 60 sec  (default: 60 sec)
  Short-path-forwarding: Enabled
  Revert-Priority: unset; SPF Reverted: No
  Hold time: 0 sec  (default: 0 sec)
  Master Down interval: 4 sec
  Trackport:
    Port(s)                   Priority  Port Status
    =======                   ========  ===========

  Global Statistics:
  ==================
    Checksum Error : 0
    Version Error  : 0
    VRID Invalid   : 0

  Session Statistics:
  ===================
    Advertisements             : Rx: 211, Tx: 5111
    Neighbor Advertisements              : Tx: 2556
    Session becoming master    : 1
    Advts with wrong interval  : 0
    Prio Zero pkts             : Rx: 0, Tx: 0
    Invalid Pkts Rvcd          : 0
    Bad Virtual-IP Pkts        : 0
    Invalid Authenticaton type : 0
    Invalid TTL Value          : 0
    Invalid Packet Length      : 0
    VRRPE backup advt sent     : 3
    VRRPE backup advt recvd    : 84
```

3. Enter the **clear ipv6 vrrp statistics** command.

```
device# clear ipv6 vrrp statistics
```

4. Enter the **show ipv6 vrrp detail** command to confirm the changes.

```
device# show ipv6 vrrp detail

================================================================================
Rbridge-id:14
================================================================================

Total number of VRRP session(s)   : 1

VRID 11
  Interface: Ve 100;  Ifindex: 1207959652
  Mode: VRRP
  Admin Status: Enabled
  Description :
  Address family: IPv6
  Version: 3
  Authentication type: No Authentication
  State: Master
  Session Master IP Address: Local
  Backup Router(s): fe80::205:33ff:fe65:9d44
  Virtual IP(s): fe80::1
  Virtual MAC Address: 02e0.5200.250b
  Configured Priority: unset (default: 100); Current Priority: 100
  Advertisement interval: 1 sec  (default: 1 sec)
  Preempt mode: ENABLE  (default: DISABLED)
  Advertise-backup: ENABLE  (default: DISABLED)
  Backup Advertisement interval: 60 sec  (default: 60 sec)
  Short-path-forwarding: Enabled
  Revert-Priority: unset; SPF Reverted: No
  Hold time: 0 sec  (default: 0 sec)
  Master Down interval: 4 sec
  Trackport:
    Port(s)                   Priority  Port Status
    =======                   ========  ===========

  Global Statistics:
  ==================
    Checksum Error : 0
    Version Error  : 0
    VRID Invalid   : 0

  Session Statistics:
  ===================
    Advertisements             : Rx: 0, Tx: 3
    Neighbor Advertisements              : Tx: 1
    Session becoming master    : 0
    Advts with wrong interval  : 0
    Prio Zero pkts             : Rx: 0, Tx: 0
    Invalid Pkts Rvcd          : 0
    Bad Virtual-IP Pkts        : 0
    Invalid Authenticaton type : 0
    Invalid TTL Value          : 0
    Invalid Packet Length      : 0
    VRRPE backup advt sent     : 0
    VRRPE backup advt recvd    : 0
```

In this show output you can see that the statistical counters have been reset. Although some of the counters show numbers because IPv6 VRRP traffic is still flowing, the numbers are much lower than in the initial **show** command output.

VRRP-Ev3 Overview

VRRP Extended version 3 (VRRP-Ev3) introduces IPv6 address support to the Extreme Networks proprietary VRRP Extended version 2 (VRRP-Ev2) protocol. VRRP-Ev3 is designed to avoid the limitations in the standards-based VRRPv3 protocol.

To create VRRP-Ev3, Extreme Networks has implemented the following differences from the RFC 5798 that describes VRRPv3 to provide extended functionality and ease of configuration:

- VRRP-Ev3 does not include the concept of an owner device and a master VRRP-Ev3 device is determined by the priority configured on the device.
- While the VRRP-Ev3 virtual router IP address must belong in the same subnet as a real IP address assigned to a physical interface of the device on which VRRP-Ev3 is configured, it must not be the same as any of the actual IP addresses on any interface.
- Configuring VRRP-Ev3 uses the same task steps for all devices; no differences between master and backup device configuration. The device configured with the highest priority assumes the master role.

  NOTE
  VRRP-Ev3 is supported on the devices described in this guide. In a mixed-device environment, consult your documentation for the other devices to determine if VRRP-Ev3 is supported.
  VRRP-Ev3 does not interoperate with with VRRPv2 or VRRPv3 sessions.

Enabling IPv6 VRRP-Ev3

IPv6 VRRP-Ev3 is enabled on a device when a virtual IPv6 address is assigned to a VRRP-Ev3 group.

Before assigning a virtual IPv6 address to an IPv6 VRRPv3 group, you must configure IPv6 VRRP-Ev3 on a virtual ethernet interface and assign a VRRPv3 group to the device. The IPv6 VRRP-Ev3 session is enabled after the configuration of an IPv6 virtual IP address. The configuration example following after the individual steps represents all the steps together in order.

1. Enter the **configure** command to access the global configuration mode.

   ```
   device# configure
   ```

2. Enter the **rbridge-id** command with an RBridge ID to enter RBridge ID configuration mode.

   ```
   device(config)# rbridge-id 122
   ```

3. To globally enable VRRP-Ev3, enter the **ipv6 protocol vrrp-extended** command.

   ```
   device(config-rbridge-id-122)# ipv6 protocol vrrp-extended
   ```

4. Enter the **interface ve** command with an associated virtual Ethernet (VE) interface number.

   ```
   device(config-rbridge-id-122)# interface ve 2019
   ```

   In this example, virtual Ethernet (ve) configuration mode is entered and the interface is assigned with a VE number of 2019.

5. Enter an IPv6 address for the interface using the **ipv6 address** command.

   ```
   device(config-if-ve-2019)# ipv6 address 2001:2019:8192::122/64
   ```

6. Enter the **ipv6 vrrp-extended-group** command with a number to assign a VRRP-E group to the device.

   ```
   device(config-if-ve-2018)# ipv6 vrrp-extended-group 19
   ```

   In this example, VRRP-Ev3 group configuration mode is entered.

7. Enter the **virtual-ip** command to assign a link-local virtual IPv6 address to a VRRPv3 group.

```
device(config-vrrp-extended-group-19)# virtual-ip fe80::2019:1
```

   In this example, the IPv6 address of the virtual router is assigned to VRRP-Ev3 group 19 and the VRRP-Ev3 session is enabled.

   NOTE
   A maximum of two virtual IPv6 addresses can be configured on VRRP-Ev3 group. For VRRPv3, Extreme recommends using two IPv6 addresses; one link local address and one global address.

8. Enter the **virtual-ip** command to assign a virtual IPv6 address to a VRRPv3 group.

```
device(config-vrrp-extended-group-19)# virtual-ip 2001:2019:8192::1
```

   In this example, a global IPv6 address is configured for the virtual router.

The following example shows how to enable a VRRP-E-v3 session by assigning a virtual IP address to an extended VRRP-E-v3 virtual group.

```
device# configure
device(config)# rbridge-id 122
device(config-rbridge-id-122)# ipv6 protocol vrrp-extended
device(config-rbridge-id-122)# interface ve 2019
device(config-ve-2019)# ipv6 address 2001:2019:8192::122/64
device(config-ve-2019)# ipv6 vrrp-extended-group 19
device(config-vrrp-extended-group-19)# virtual-ip fe80::2019:1
device(config-vrrp-extended-group-19)# virtual-ip 2001:2019:8192::1
```

After enabling a VRRP-Ev3 session, you may need to configure some optional parameters such as short-path forwarding for load-balancing or tracking an interface.

VRRP-E load-balancing using short-path forwarding

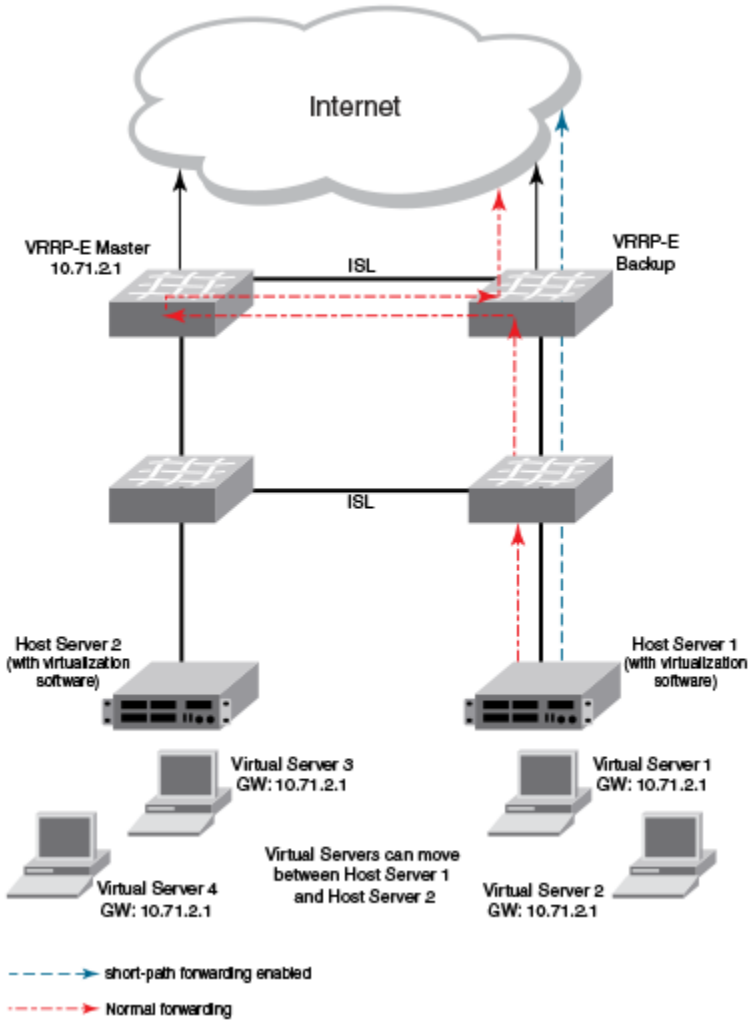
The VRRP-E Extension for Server Virtualization feature allows devices to bypass the VRRP-E master router and directly forward packets to their destination through interfaces on the VRRP-E backup router. This is called *short-path forwarding*. A backup router participates in a VRRP-E session only when short-path forwarding is enabled.

VRRP-E active-active load-balancing is available in VCS mode and uses flow-hashing techniques to determine the path. All nodes in the VCS are aware of all VRRP-E sessions and the participating RBridges in each session.

Packet routing with short-path forwarding to balance traffic load

When short-path forwarding is enabled, traffic load-balancing is performed because both master and backup devices can be used to forward packets.

FIGURE 57 Short-path forwarding



If you enable short-path forwarding in both master and backup VRRP-E devices, packets sent by Host Server 1 (in the figure) and destined for the Internet cloud through the device on which a VRRP-E backup interface exists can be routed directly to the VRRP-E backup device (blue dotted line) instead of being switched to the master router and then back (red dotted-dash line).

In the figure, load-balancing is achieved using short-path forwarding by dynamically moving the virtual servers between Host Server 1 and Host Server 2.

Short-path forwarding with revert priority

Revert priority is used to dynamically enable or disable VRRP-E short-path forwarding.

If short-path forwarding is configured with revert priority on a backup router, the revert priority represents a threshold for the current priority of the VRRP-E session. When the backup device priority is higher than the configured revert priority, the backup router is able to perform short-path forwarding. If the backup priority is lower than the revert priority, short-path forwarding is disabled.

Configuring VRRP-Ev3 load-balancing

VRRP-Ev3 traffic can be load-balanced using short-path forwarding on the backup devices.

Before configuring VRRP-Ev3 load-balancing, VRRP-Ev3 must be configured on all devices in the VRRP-Ev3 session.

Perform this task on all backup VRRP-Ev3 Layer 3 devices to allow load sharing within an IPv6 VRRP extended group.

1. Use the **configure terminal** command to enter global configuration mode.

   ```
   device# configure terminal
   ```

2. Enter the **rbridge-id** command with an associated RBridge ID to enter RBridge configuration mode for a specific ID.

   ```
   device(config)# rbridge-id 122
   ```

3. Globally enable VRRP-Ev3.

   ```
   device(config-rbridge-id-122)# ipv6 protocol vrrp-extended
   ```

4. Enter the **interface ve** command with an associated VLAN number.

   ```
   device(config-rbridge-id-122)# interface ve 2019
   ```

   In this example, virtual Ethernet (ve) configuration mode is entered and the interface is assigned with a VLAN number of 2019.

5. Enter an IPv6 address for the interface using the **ipv6 address** command.

   ```
   device(config-ve-2019)# ipv6 address 2001:2019:8192::122/64
   ```

6. Enter the **ipv6 vrrp-extended-group** command with a number to assign a VRRP-E group to the device.

   ```
   device(config-ve-2018)# ipv6 vrrp-extended-group 19
   ```

   In this example, VRRP-Ev3 group configuration mode is entered.

7. Enter the **short-path-forwarding** command with a **revert-priority** value to configure the backup VRRP-E as an alternate path with a specified priority.

   ```
   device(config-vrrp-extended-group-19)# short-path-forwarding revert-priority 50
   ```

   When the backup device priority is higher than the configured **revert-priority** value, the backup router is able to perform short-path forwarding. If the backup priority is lower than the revert priority, short-path forwarding is disabled.

In the following example, short-path forwarding is configured on a backup VRRP-Ev3 device and a revert priority threshold is configured. If the backup device priority falls below this threshold, short-path forwarding is disabled.

```
device# configure
device(config)# rbridge-id 122
device(config-rbridge-id-122)# ipv6 protocol vrrp-extended
device(config-rbridge-id-122)# interface ve 2019
device(config-ve-2019)# ipv6 address 2001:2019:8192::122/64
device(config-ve-2019)# ipv6 vrrp-extended-group 19
device(config-vrrp-extended-group-19)# short-path-forwarding revert-priority 50
```

Displaying and clearing VRRP-Ev3 statistics

Several show commands can display statistical information about IPv6 VRRP-Ev3 configurations. To reset the IPv6 VRRP-Ev3 statistics, there is a CLI command.

Before displaying statistics, VRRP-Ev3 must be configured and enabled in your network to generate traffic.

Use one or more of the following commands to display VRRP-Ev3 information. The commands do not have to be entered in this order.

1. Use the **exit** command to return to privileged EXEC mode, if required.
2. Enter the **show ipv6 vrrp-extended** command.

```
device# show ipv6 vrrp-extended

============Rbridge-id:122============

Total number of VRRP session(s)   : 1

VRID 19
  Interface: Ve 2019;  Ifindex: 1207961571
  Mode: VRRPE
  Admin Status: Enabled
  Description :
  Address family: IPv6
  Version: 3
  Authentication type: No Authentication
  State: Backup
  Session Master IP Address: fe80::205:33ff:fe79:fb1e
  Virtual IP(s): 2001:2019:8192::1
  Configured Priority: unset (default: 100); Current Priority: 100
  Advertisement interval: 1 sec  (default: 1 sec)
  Preempt mode: DISABLE  (default: DISABLED)
  Advertise-backup: ENABLE  (default: DISABLED)
  Backup Advertisement interval: 60 sec  (default: 60 sec)
  Short-path-forwarding: Enabled
  Revert Priority: unset; SPF reverted: No
  Hold time: 0 sec  (default: 0 sec)
  Trackport:
    Port(s)                   Priority  Port Status
    =======                   ========  ===========
  Statistics:
    Advertisements: Rx: 102992, Tx: 1716
    Neighbor Advertisements: Tx: 0
```

This example output shows that one IPv6 VRRP-E session is configured.

3. Enter the **show ipv6 vrrp summary** command.

```
device# show ipv6 vrrp summary

============Rbridge-id:122============

Total number of VRRP session(s)   : 2
Master session count  : 1
Backup session count  : 1
Init session count    : 0

VRID  Session  Interface  Admin    Current   State   Short-path  Revert    SPF
                          State    Priority          Forwarding  Priority  Reverted
====  =======  =========  =====    ========  =====   ==========  ========  ========
18    VRRPE    Ve 2018    Enabled  254       Master  Enabled     unset     No
19    VRRPE    Ve 2019    Enabled  100       Backup  Enabled     unset     No
```

This example shows summary output for the two IPv6 VRRP-E sessions that are configured for virtual routers 18 and 19.

4. Enter the **show ipv6 vrrp 19 detail** command.

```
device# show ipv6 vrrp 19 detail

============Rbridge-id:122============

Total number of VRRP session(s)   : 1

VRID 19
  Interface: Ve 2019;  Ifindex: 1207961571
  Mode: VRRPE
  Admin Status: Enabled
  Description :
  Address family: IPv6
  Version: 3
  Authentication type: No Authentication
  State: Backup
  Session Master IP Address: fe80::205:33ff:fe79:fb1e
  Virtual IP(s): 2001:2019:8192::1
  Virtual MAC Address: 02e0.5200.2513
  Configured Priority: unset (default: 100); Current Priority: 100
  Advertisement interval: 1 sec  (default: 1 sec)
  Preempt mode: DISABLE  (default: DISABLED)
  Advertise-backup: ENABLE  (default: DISABLED)
  Backup Advertisement interval: 60 sec  (default: 60 sec)
  Short-path-forwarding: Enabled
  Revert-Priority: unset; SPF Reverted: No
  Hold time: 0 sec  (default: 0 sec)
  Master Down interval: 4 sec
  Trackport:
    Port(s)                   Priority  Port Status
    =======                   ========  ===========

  Global Statistics:
  ==================
    Checksum Error : 0
    Version Error  : 0
    VRID Invalid   : 0

  Session Statistics:
  ===================
    Advertisements             : Rx: 103259, Tx: 1721
    Neighbor Advertisements              : Tx: 0
    Session becoming master    : 0
    Advts with wrong interval  : 0
    Prio Zero pkts             : Rx: 0, Tx: 0
    Invalid Pkts Rvcd          : 0
    Bad Virtual-IP Pkts        : 0
    Invalid Authenticaton type : 0
    Invalid TTL Value          : 0
    Invalid Packet Length      : 0
    VRRPE backup advt sent     : 1721
    VRRPE backup advt recvd    : 0
```

This example shows detailed output for the IPv6 VRRP-E session for virtual router 19.

5. Enter the **clear ipv6 vrrp statistics** command with the **all** option to reset the statistical counters for all IPv6 VRRP-Ev3 sessions.

```
device# clear ipv6 vrrp statistics all
```

6. Enter the **clear ipv6 vrrp statistics** command with the **session** option to reset the statistical counters for the IPv6 VRRP-Ev3 session for virtual router 19.

```
device# clear ipv6 vrrp statistics session 19
```

Fabric-Virtual-Gateway

Fabric-Virtual-Gateway overview

Fabric-Virtual-Gateway is an Extreme-proprietary implementation of a router redundancy protocol that enables VCS Fabrics to support scalable Layer 3 gateway configuration requirements.

The Fabric-Virtual-Gateway feature allows multiple RBridges in a VCS Fabric to form a group of gateway routers and share the same gateway IP address for a given subnet. The gateway IP address is similar to a virtual IP address in VRRP terminology. The gateway IP address can be configured on all the nodes forming a Fabric-Virtual-Gateway group without the need to assign a unique IP address on each of the RBridges for a given subnet. Only one gateway IP address is allowed per subnet.

Fabric-Virtual-Gateway does not exchange any advertisements or keepalive frames over the network to or from the group, or detect when the RBridge acting as an ARP responder is down. Instead, it leverages the Extreme-proprietary VCS Fabric services to exchange Fabric-Virtual-Gateway group information among the participating nodes.

Fabric-Virtual-Gateway is configured on a global virtual Ethernet (VE) interface so that all participating nodes forming the Fabric-Virtual-Gateway redundant group have the same gateway address.

Fabric-Virtual-Gateway provides the following:

- A mechanism to configure IPv4 or IPv6 gateways for each Layer 3 VE interface in a VCS Fabric
- Active-active gateway load-balancing
- A mechanism that does not require configuring a primary IP address on the Layer 3 interface to enable gateway functionality
- Support for configuring a large number of Layer 3 gateways, and the capability for future expansion

Within a Fabric-Virtual-Gateway group, only one participant RBridge for a given subnet responds to ARP requests for the gateway IP address. The Fabric-Virtual-Gateway group master (ARP responder) is elected once during the configuration, and the master takes the responsibility for responding to ARP requests. The Fabric-Virtual-Gateway group master does not change dynamically and is elected only when the existing master leaves the group or VCS Fabric.

There is no explicit configuration to elect a node as the ARP responder for the gateway IP address.

Fabric-Virtual-Gateway limitations

Fabric-Virtual-Gateway has the following limitations:

- Fabric-Virtual-Gateway (FVG) works on VDX platforms only and does not work on any other platform or with other vendors' products.
- FVG is not supported across VCS Fabrics. It is supported only under logical chassis/management cluster modes.
- FVG is not compatible with VRRP or VRRP-E, and does not replace or make VRRP or VRRP-E obsolete. VRRP or VRRP-E continues to function on VDX platforms with the present scaling limits.

- A real IP address is required to get a ping response from VCS Fabric hosts in a FVG configuration. A real IP address is required on a VE interface for communication that requires two-way exchanges of packets directly with the VDX itself on that VE interface. This includes, but is not necessarily limited to, SSH, SCP, FTP, SNMP queries, NTP, DNS, and TFTP. This limitation applies if the VDX is receiving incoming communications (for example, SSH logins or snmpwalk) or is initiating outgoing communications (for example, during a supportsave upload).
- DHCP relay is not supported in conjunction with the FVG on the same VE interface.
- DHCP relays do not function with FVG feature alone. The **ip dhcp relay address** command must have an IPv4 address configured in order for the RBridge ID VE to function properly.

Gateway behavior per RBridge

You can configure load-balancing, minimum priority for a gateway and set the gratuitous ARP timer per RBridge.

- Load balancing

  The ARP responder normally routes the traffic. Once the load balancing threshold priority value is exceeded, the device in the network routes the traffic along with the ARP responder.

  Load balancing for the gateway virtual MAC address is enabled by default. However, you can disable load balancing.

  Load balancing behavior can be controlled on a per-session basis.

- Interface tracking

  Multiple interfaces can be tracked for a given Fabric-Virtual-Gateway session. In tracking the state of the interface is tracked (absent , up or in down state).

  The threshold priority for acting as a gateway is tracked. If the sum of track priority (the priority associated with the element being tracked) goes below the threshold priority specified for load balancing, the router relinquishes the ARP responder role and re-election occurs. If the router was not elected to be the ARP responder, and load balancing with threshold priority was configured, the router will not participate in active-active load balancing.

- Network or host tracking

  Similar to interface tracking, you can track the availability of a route or the reachability of a destination and can specify the priority associated with each tracked entity.

  The next-hop tracking checks for the reachability of the destination.

- Gratuitous ARP

  By default, periodic gratuitous ARP is disabled for Fabric-Virtual-Gateway.

  You can start sending gratuitous ARP packets periodically for all sessions from the Fabric-Virtual-Gateway and enable periodic gratuitous ARP for a specific session.

Fabric-Virtual-Gateway configuration notes

Enabling Fabric-Virtual-Gateway in the VCS Fabric

Fabric-Virtual-Gateway must first be enabled in the VCS Fabric, by means of the **router fabric-virtual-gateway** command, before it can be configured on individual VE interfaces, as in the following example:

```
device(config)# router fabric-virtual-gateway
device(config-router-fabric-virtual-gateway)#
```

By default, both IPv4 and IPv6 address families are enabled in this way. To disable this, use the **no enable** command in IPv4 or IPv6 address family configuration mode, respectively, as in the following examples.

```
device(config)# router fabric-virtual-gateway
device(config-router-fabric-virtual-gateway)# address-family ipv4
device(config-address-family-ipv4)# no enable

device(config)# router fabric-virtual-gateway
device(config-router-fabric-virtual-gateway)# address-family ipv6
device(config-address-family-ipv6)# no enable
```

When the **no enable** command is issued, the Fabric-Virtual-Gateway configuration for the specified address family is deactivated on all RBridges in the VCS Fabric. Use the **enable** command in the above configuration modes to reactivate Fabric-Virtual-Gateway sessions.

Global VE configuration

The global VE interface mode acts as a template to hold the VE-specific configuration in the VCS Fabric. However, it does not provision the VLAN or create a Layer 3 VE interface on the individual RBridges automatically. Before a global VE can be configured, the corresponding VLAN must be configured. (Ranging is supported.)

The following example creates a global VE interface:

```
 device(config)# interface vlan 5124
device(config-Vlan-5124)# exit
device(config)# interface ve 5124
device(config-Ve-5124)#
```

Once a global VE interface is created, it can be attached to individual RBridges by means of the **attach** command, as in the following example. Deleting the global VE interface also deletes the corresponding RBridge-level VE interface if it is configured.

```
device(config)# interface ve 5124
device(config-Ve-5124)# attach rbridge-id add 1,2,10
```

Attaching a global VE to one or more RBridges, as in the following example, provisions the VLAN on those Rbridges and automatically creates a Layer 3 VE interface on the specified RBridges. The **remove** keyword under the **attach** command detaches the VE interface from one or more RBridges in the VCS Fabric.

```
device(config)# interface Ve 5124
device(config-interface-ve)# attach rbridge-id remove 1,2
```

Once RBridges are attached to the global VE interface, the administrative enable and disable operation is controlled only from the global VE interface configuration mode. The global VE interface can be administratively disabled and enabled by means of regular **shutdown** and **no shutdown** commands under the global VE mode, as in the following example.

```
device(config)# interface ve 5124
device(config-Ve-5124)# shutdown
device(config-Ve-5124)# no shutdown
```

RBridge-level VE configuration

The Fabric-Virtual-Gateway configuration is available under the RBridge VE interface to control some of the parameters of the RBridge. The RBridge-level VE interface configuration works the same way as it does in previous releases. Even if a global VE interface is attached to a specific RBridge, the user can administratively enable or disable the VE interface under the RBridge, with no effect on the global VE interface. If the user has already created an RBridge-level VE interface, and later the global VE interface is attached to the RBridge, the global VE interface shut/no-shut configuration overwrites the RBridge-level configuration. When a global VE interface is either removed or detached from the RBridge, the RBridge-level VE interface is deleted and all user configuration under the RBridge-level VE interface is lost if that interface does not have an IPv4 or IPv6 address configured.

The RBridge-level VE interface configurations, such as assigning IPv4/IPv6 addresses, proxy ARP, MTU, and so on, are still allowed even when an RBridge is attached to a global VE interface.

Layer 3 Fabric-Virtual-Gateway

Similar to conventional VRRP/VRRP-E, the gateway address for a given subnet must be the same on all of the nodes in the VCS Fabric that participate in the Fabric-Virtual-Gateway group. In the case of VRRP/VRRP-E, this validation is user-driven, and in case of a misconfiguration multiple gateway routers can claim to be the default gateway. With the advent of a global VE interface, and the availability of gateway IP address configuration only under the global VE interface, the scope of manual error is reduced.

A Fabric-Virtual-Gateway configuration, by means of the **ip fabric-virtual-gateway** or **ipv6 fabric-virtual-gateway** commands under a global VE interface, acts as template, and is activated only for those RBridges to which the global VE interface is attached. The following example is a typical Fabric-Virtual-Gateway configuration for both IPv4 and IPv6.

```
device(config)# interface ve 6000
device(config-Ve-6000)# attach rbridge-id 10,11,12
device(config-Ve-6000)# no shutdown
device(config-Ve-6000)# ip fabric-virtual-gateway
device(config-ip-fabric-virtual-gateway )# gateway-address 10.65.40.100/24
device(config-ip-fabric-virtual-gateway )# exit
device(config-Ve-6000)# ipv6 fabric-virtual-gateway
device(config-ipv6-fabric-virtual-gateway )# gateway-address fe80::400/64
```

Multiple gateway IP addresses for IPv4 Fabric-Virtual-Gateway

Network OS supports multiple gateway IP addresses for IPv4 Fabric-Virtual-Gateway (FVG).

Gateway IPs from multiple subnets (maximum of 32) can be configured for each FVG session. Multiple gateway IPs from the same subnet can be configured, but the number of FVG sessions for each interface remains one. A single RBridge becomes the ARP responder for all the gateway IPs configured for the session.

Multiple gateway IPs are supported only for IPv4.

All restrictions for configuring an FVG gateway applies to multiple gateway IP addresses as well. If IP conflicts are detected for any gateway IP configured on the session, the configuration is accepted with a RASLOG, but the session is invalidated until the conflict is resolved.

Periodic gratuitous address resolution protocol (GARP), if configured, would be sent out only for the first gateway address. When a session moves to Master, GARP is sent out for all Gateway IP addresses configured on the session.

When downgrading to earlier versions of Network OS, if multiple gateway IPs are present then all gateway IP configurations are removed after downgrade. If only one gateway IP present, then it is retained.

Gateway MAC address

The gateway MAC address is used to respond to ARP requests for a gateway IP address. The default IPv4 MAC address is 02e0.5200.01ff. The default IPv6 MAC address is 02e0.5200.02fe.

When Fabric-Virtual-Gateway is configured on multiple VCS Fabrics connected to each other, by default all of the VCS Fabrics will use the same gateway MAC address for all of the Fabric-Virtual-Gateway sessions. This leads to multiple IP address being resolved to the same MAC address. To avoid this, it is necessary to specify a different gateway MAC address for both IPv4 and IPv6 Fabric-Virtual-Gateway sessions in the VCS Fabric. The gateway MAC address can be changed globally for IPv4 and IPv6 address families, as in the following example.

```
device(config)# router fabric-virtual-gateway
device(config-router-fabric-virtual-gateway)# address-family ipv4
device(config-address-family-ipv4)#  gateway-mac-address 00a0.b100.2000
device(config-address-family-ipv4)# exit
device(config-router-fabric-virtual-gateway)# address-family ipv6
device(config-address-family-ipv6)# gateway-mac-address 0220.5e00.0012
```

Note the following considerations:

- The lowest and highest 24 bits cannot be all zeroes in a user-defined gateway MAC address.
- Instead of specifying an arbitrary gateway MAC address, the user can pick an address from the VRRP-E range of MAC address (for example, 02e0.5200.00*xx*, where *xx* value be used to differentiate each VCS Fabric.
- Before configuring or unconfiguring a user-defined gateway MAC address, the address family must be disabled administratively. The address family can be disabled by means of the **no enable** command under Fabric-Virtual-Gateway address-family configuration mode.

Enabling or disabling Fabric-Virtual-Gateway sessions on an RBridge

The administrative state of a session can be flipped on a per-RBridge basis, as it might be necessary to disable or re-enable the Fabric-Virtual-gateway session on selected RBridges.

The following example disables Fabric-Virtual-Gateway for IPv4 address family on an RBridge.

```
device(config)# rbridge-id 2
device(config-rbridge-id-2)# interface ve 6000
device(config-Ve-6000)# ip fabric-virtual-gateway
device(config-ip-fabric-virtual-gateway)# disable
```

Use the **no** form of the **disable** command to remove RBridge-level preference. Similarly, when a session is administratively disabled at the global VE interface level, use the **enable** command under RBridge-level Fabric-Virtual-Gateway configuration mode to enable a sessions on the RBridge. The **no** form of the **disable** and **enable** commands removes the RBridge-level preference with respect to the administrative state as in the following example.

```
device(config)# rbridge-id 2
device(config-rbridge-id-2)# interface ve 6000
device(config-Ve-6000)# ip fabric-virtual-gateway
device(config-ip-fabric-virtual-gateway)# no disable
device(config-ip-fabric-virtual-gateway)# no enable
```

When neither a disable or enable configuration exists under RBridge-level Fabric-Virtual-Gateway configuration mode, the administrative state of the session is determined by the global VE interface configuration for Fabric-Virtual-Gateway.

Behaviors specific to Fabric-Virtual-Gateway

The following behaviors are specific to Fabric-Virtual-Gateway:

- Only one Fabric-Virtual-Gateway session is allowed under a VE interface.
- Only one gateway address can be specified for a Fabric-Virtual-Gateway session. With IPv6, only one global scope and one link-local scope address are allowed.
- Active-active load balancing is enabled by default. However, it can be disabled globally.
- There is no concept of master or backup in active-active configuration. One of the Fabric-Virtual-Gateway member routers is elected as the default ARP responder.
- Only the ARP responder responds to ARP requests for the gateway IP address.
- In case active-active load balancing is disabled, only the elected ARP responder acts as the gateway router, while the remaining Fabric-Virtual-Gateway members forward the traffic to the ARP responder.
- On all of the RBridges where Fabric-Virtual-Gateway configuration is not activated or applied, the gateway MAC address entry is installed in the hardware to load-balance or forward all gateway traffic to the Fabric-Virtual-Gateway members.
- The ARP responder is re-elected when the existing ARP responder exits the VCS Fabric or the Fabric-Virtual-Gateway configuration is detached from the RBridge.
- The ARP responder may be re-elected when active-active load balancing is disabled on an existing ARP responder.

Fabric-Virtual-Gateway configuration

The Fabric-Virtual-Gateway configuration consists of three tasks:

1. Enabling router Fabric-Virtual-Gateway

   The Fabric-Virtual-Gateway protocol must be enabled in the router address-family before it can be configured on individual VE interfaces.

2. Configuring Fabric-Virtual-Gateway on a global VE interface

   The following configurations are allowed under global VE interface mode only:

   - Gateway IP address
   - Periodic gratuitous ARP control
   - Hold time
   - Load balancing across all RBridges

3. Attaching a global VE interface to specific RBridge(s), and changing the administrative state of the global VE interface.
4. Configuring Fabric-Virtual-Gateway in an RBridge VE interface

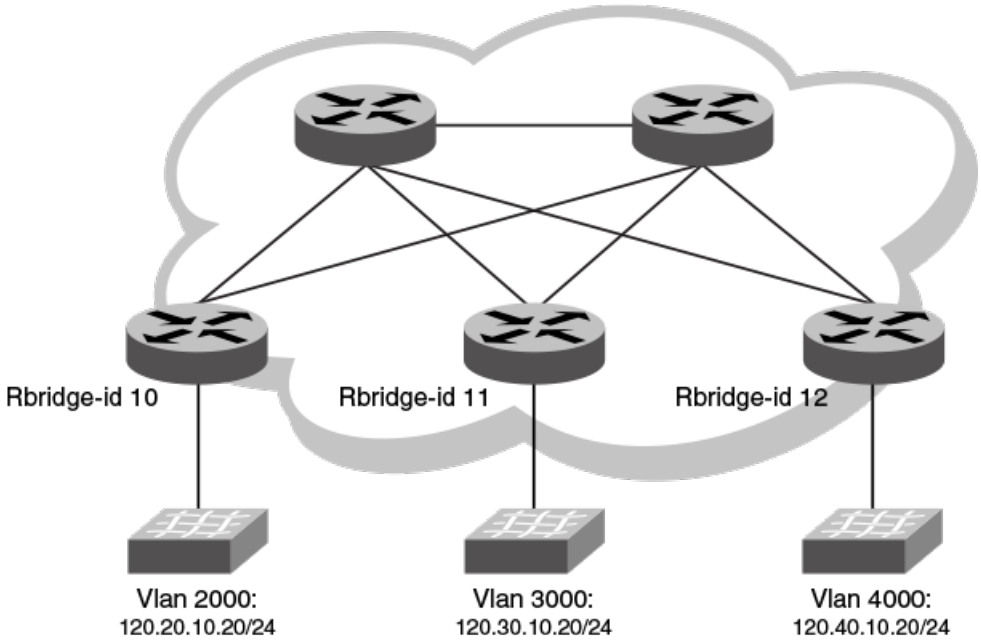
   The following configurations are allowed under the VE interface on an RBridge only:

   - Load-balancing threshold priority
   - Interface tracking
   - Route or next hop tracking

   For a Fabric-Virtual-Gateway configuration to be applicable to all or a set of RBridges, the global VE interface must be attached to all or a set of RBridges, respectively. The RBridge-level Fabric-Virtual-Gateway configuration is not required unless interface, route, or nexthop tracking is needed.

The following illustrations depicts sample configurations.

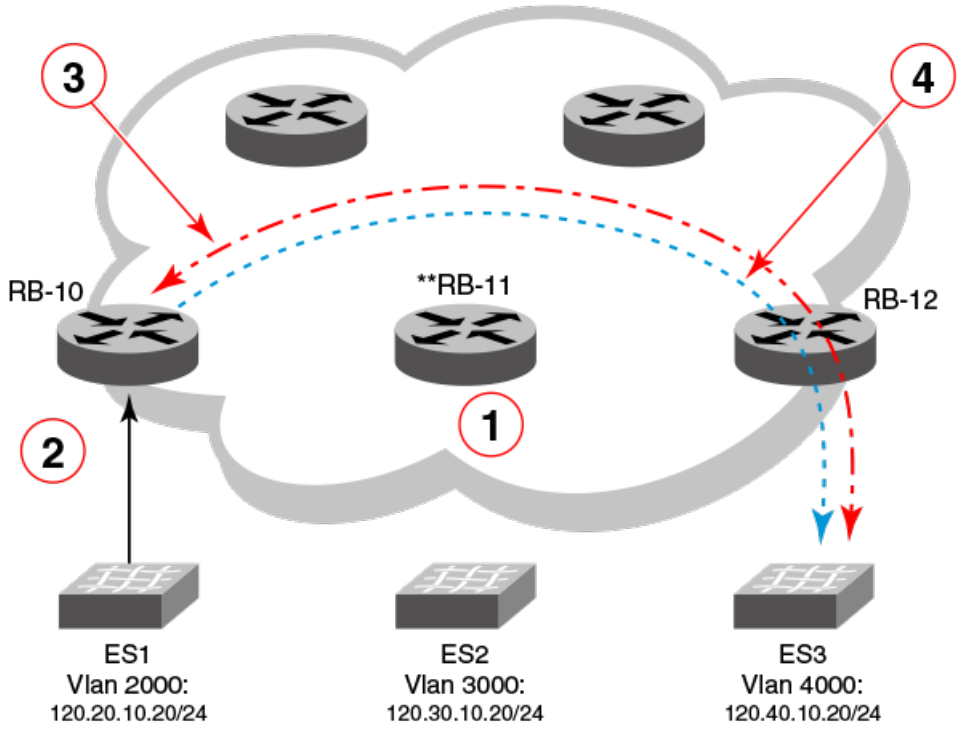
FIGURE 58 East-west intersubnet traffic flows under homogenous subnet configuration: Example 1



Sample Configuration for the above illustration:

```
Interface Ve 2000
  attach add rbridge 10, 11, 12
  ip fabric-virtual gateway
  gateway-address 120.20.10.10/24
Interface Ve 3000
  attach add rbridge 10, 11, 12
  ip fabric-virtual gateway
  gateway-address 120.30.10.10/24
Interface Ve 4000
  attach add rbridge 10, 11, 12
  ip fabric-virtual gateway
  gateway-address 120.40.10.10/24
```

FIGURE 59 East-west intersubnet traffic flows under homogenous subnet configuration: Example 2



1. End-station ES1 resolves ARP for gateway IP address, which is resolved by ARP responder **RB-11.
2. Once ARP for gateway IP address is resolved, end-station ES1 sends data traffic with destination IP 120.40.10.20. RB-10 routes the frames once ARP for destination ES3 is resolved in slow path.
3. Shows slow-path ARP learning triggered by RB-10, and ARP for 120.40.10.20 resolved at ingress RBridge RB-10.
4. Shows Layer 3 data traffic getting TRILL switched within VCS and reaching RB-12 and the destination.

Enabling and configuring Fabric-Virtual-Gateway globally (IPv4)

To enable the Fabric-Virtual-Gateway configuration globally, perform the following steps.

1. Enter the **router fabric-virtual-gateway** command in global configuration mode.

```
device(config)# router fabric-virtual-gateway
```

2. Enter the **address-family ipv4** command to configure the IPv4 address-family.

```
device(conf-router-fabric-virtual-gateway)# address-family ipv4
```

NOTE

The Fabric-Virtual-Gateway IPv4 configuration is automatically enabled when the IPv4 address family is configured. If the Fabric-Virtual-Gateway IPv4 configuration is disabled, enter the **enable** command to enable it.

3. (Optional) Enter the **gateway-mac-address** command to assign a gateway MAC address to the IPv4 Fabric-Virtual-Gateway.

   NOTE
   If **gateway-mac-address** is not configured, then the device takes the default gateway MAC address. The default gateway MAC address for IPv4 is 02e0.5200.01ff.

```
device(conf-address-family-ipv4)# gateway-mac-address 0011.2222.2233
```

4. (Optional) Enter the **gratuitous-arp** command to set the ARP timer.

```
device(conf-address-family-ipv4)# gratuitous-arp timer 60
```

5. (Optional) Enter the **accept-unicast-arp-request** command to respond to unicast ARP request.

```
device(conf-address-family-ipv4)# accept-unicast-arp-request
```

6. (Optional) In privileged EXEC mode, enter the **show ip fabric-virtual-gateway detail** command to view the configured settings.

```
device# show ip fabric-virtual-gateway detail
```

The following example shows how to configure IPv4 Fabric-Virtual-Gateway.

```
device(config)# router fabric-virtual-gateway
device(conf-router-fabric-virtual-gateway)# address-family ipv4
device(conf-address-family-ipv4)# gateway-mac-address 0011.2222.2233
device(conf-address-family-ipv4)# gratuitous-arp timer 60
device(conf-address-family-ipv4)# accept-unicast-arp-request
```

The following example shows the output of the **show ip fabric-virtual-gateway detail** command.

```
============Rbridge-id:77============
Total number of IPv4 Fabric Virtual Gateway sessions   : 1
Total number of sessions in Active state   : 1
Total number of sessions in InActive state : 0
Total number of sessions in Init state     : 0

Interface: Ve 100;  Ifindex: 1207959652
  Admin Status: Enabled
  Description :
  Address family: IPV4    State: Active
  ARP responder Rbridge-id:  77
  Gateway IP: 1.1.1.1/24
  Gateway MAC Address: 02e0.5200.01ff
  Load balancing configuration: Enabled
  Load balancing current status: Enabled
  Load balancing threshold priority: unset
  Gratuitous ARP Timer: Disabled
  Hold time: 0 sec  (default: 0 sec)
    Total no. of state changes: 1
    Gratuitous ARP Sent: 1
  Last state change: 0d.0h.1m.5s ago
  Track Priority: 0
```

Enabling and configuring Fabric-Virtual-Gateway globally (IPv6)

To enable Fabric-Virtual-Gateway configuration globally, perform the following steps.

1. Enter the **router fabric-virtual-gateway** command in global configuration mode.

```
device(config)# router fabric-virtual-gateway
```

2. Enter the **address-family ipv6** command to configure the IPv6 address-family.

```
device(conf-router-fabric-virtual-gateway)# address-family ipv6
```

NOTE
The Fabric-Virtual-Gateway IPv6 configuration is automatically enabled when the IPv6 address family is configured. If the Fabric-Virtual-Gateway IPv6 configuration is disabled, enter the **enable** command to enable it.

3. (Optional) Enter the **gateway-mac-address** command to assign a gateway MAC address to the IPv6 Fabric-Virtual-Gateway.

NOTE
If **gateway-mac-address** is not configured, then the device takes the default gateway MAC address. The default gateway MAC address for IPv6 is 02e0.5200.02fe.

```
device(conf-address-family-ipv6)# gateway-mac-address 0011.2222.2233
```

4. (Optional) Enter the **gratuitous-arp timer** command to set the gratuitous timer.

```
device(conf-address-family-ipv6)# gratuitous-arp timer 60
```

5. (Optional) In privileged EXEC mode, enter the **show ipv6 fabric-virtual-gateway detail** command to view the configured settings.

```
device# show ipv6 fabric-virtual-gateway detail
```

The following example shows how to configure IPv6 Fabric-Virtual-Gateway.

```
device(config)# router fabric-virtual-gateway
device(conf-router-fabric-virtual-gateway)# address-family ipv6
device(conf-address-family-ipv6)# gateway-mac-address 0011.2222.2233
device(conf-address-family-ipv6)# gratuitous-arp timer 60
```

The following example shows the output of the **show ipv6 fabric-virtual-gateway detail** command.

```
============Rbridge-id:77============
Total number of IPv6 Fabric Virtual Gateway sessions   : 1
Total number of sessions in Active state   : 1
Total number of sessions in InActive state : 0
Total number of sessions in Init state     : 0

Interface: Ve 100;  Ifindex: 1207959652
  Admin Status: Enabled
  Description :
  Address family: IPV6    State: Active
  ARP responder Rbridge-id:  77
  Gateway IP: 100::100/64
  Gateway MAC Address: 02e0.5200.02fe
  Load balancing configuration: Enabled
  Load balancing current status: Enabled
  Load balancing threshold priority: unset
  Gratuitous ARP Timer: Disabled
  Hold time: 0 sec  (default: 0 sec)
    Total no. of state changes: 1
    ND Advertisements Sent: 1
  Last state change: 0d.0h.2m.33s ago
  Track Priority: 0
```

Configuring Fabric-Virtual-Gateway on a VE interface (IPv4)

To configure Fabric-Virtual-Gateway on a VE interface, perform the following steps.

1. Enter the **interface ve** command, in global configuration mode.

```
device(config)# interface ve 2000
```

2. Enter the **attach rbridge-id add** command to add RBridge IDs to the VE interface.

```
device(config-Ve-2000)# attach rbridge-id add 54,55
```

NOTE
The VE interface is enabled in an RBridge when the **attach** command is configured in VE interface mode.
When a global VE interface is deleted, the configured RBridge-level VE interface is also deleted if no real IP address is configured in it.

NOTE
Enter the **attach rbridge-id remove** command to unattach RBridge IDs from the VE interface.

3. Enter the **ip fabric-virtual-gateway** command to configure IPv4 Fabric-Virtual-Gateway.

```
device(config-Ve-2000)# ip fabric-virtual-gateway
```

NOTE
If the Fabric-Virtual-Gateway configuration is disabled, enter the **enable** command to enable it.

4. Enter the **gateway-address** command to assign a gateway address to the IPv4 Fabric-Virtual-Gateway. Enter the gateway address in the IPv4 address/prefix format.

```
device(config-ip-fabric-virtual-gw)# gateway-address 192.128.2.1/24
```

NOTE
The **gateway-address** command installs the gateway IP address on all of the nodes in the VCS fabric to which the VE interface is attached.

5. (Optional) Enter the **hold-time** command to configure the duration, in seconds, for which the Fabric-Virtual-Gateway session is to remain idle.

```
device(config-ip-fabric-virtual-gw)# hold-time 30
```

6. (Optional) Enter the **gratuitous-arp timer** command to set the ARP timer.

```
device(config-ip-fabric-virtual-gw)# gratuitous-arp timer 15
```

7. (Optional) Enter the **load-balancing-disable** command to disable load balancing. By default, load balancing is enabled.

```
device(config-ip-fabric-virtual-gw)# load-balancing-disable
```

8. (Optional) In priviliged EXEC mode, enter the **show ip fabric-virtual-gateway interface ve** command to view the configured settings.

```
device# show ip fabric-virtual-gateway interface ve
```

The following example shows how configure an IPv4 Fabric-Virtual-Gateway on a VE interface.

```
device(config)# interface ve 2000
device(config-Ve-2000)# attach rbridge-id add 54,55
device(config-Ve-2000)# ip fabric-virtual-gateway
device(config-ip-fabric-virtual-gw)# enable
device(config-ip-fabric-virtual-gw)# gateway-address 192.128.2.1/24
device(config-ip-fabric-virtual-gw)# hold-time 30
device(config-ip-fabric-virtual-gw)# gratuitous-arp timer 15
device(config-ip-fabric-virtual-gw)# load-balancing-disable
```

The following example shows the output of the **show ip fabric-virtual-gateway interface ve** command.

```
============Rbridge-id:54============
Interface  Admin     State        Gateway         ARP        Load       Threshold   Track
           State                 IP Address      Responder  Balancing  Priority    Priority
=========  =====     =====    ================== =========  =========  =========   ==========
Ve 2000    Enabled   Active    192.128.2.1/24    Rbr-id 55  Enabled    unset       0
```

Configuring Fabric-Virtual-Gateway on a VE interface (IPv6)

To configure Fabric-Virtual-Gateway on a VE interface, perform the following steps.

1. Enter the **interface ve** command, in global configuration mode.

   ```
   device(config)# interface ve 3000
   ```

2. Enter the **attach rbridge-id add** command to add RBridge IDs to the VE interface.

   ```
   device(config-Ve-3000)# attach rbridge-id add 54-56
   ```

   NOTE
   The VE interface is enabled in an RBridge when the **attach** command is configured in VE interface mode.
   When a global VE interface is deleted, the configured RBridge-level VE interface is also deleted if no real IP address is configured in it.

   NOTE
   Enter the **attach rbridge-id remove** command to unattach RBridge IDs from the VE interface.

3. Enter the **ipv6 fabric-virtual-gateway** command to configure IPv6 Fabric-Virtual-Gateway.

   ```
   device(config-Ve-3000)# ipv6 fabric-virtual-gateway
   ```

   NOTE
   If the Fabric-Virtual-Gateway configuration is disabled, enter the **enable** command to enable it.

4. Enter the **gateway-address** command to assign a gateway address to the IPv6 Fabric-Virtual-Gateway. Enter the gateway address in the IPv6 address/prefix format.

   ```
   device(config-ipv6-fabric-virtual-gw)# gateway-address 2001:1:0:1::1/64
   ```

   NOTE
   The **gateway-address** command installs the gateway IP address on all of the nodes in the VCS fabric to which the VE interface is attached.

5. (Optional) Enter the **hold-time** command to configure the duration, in seconds, for which the Fabric-Virtual-Gateway session is to remain idle.

```
device(config-ipv6-fabric-virtual-gw)# hold-time 30
```

6. (Optional) Enter the **gratuitous-arp timer** command to set the gratuitous-arp timer.

```
device(config-ipv6-fabric-virtual-gw)# gratuitous-arp timer 15
```

7. (Optional) Enter the **load-balancing-disable** command to disable load balancing. By default, load balancing is enabled.

```
device(config-ipv6-fabric-virtual-gw)# load-balancing-disable
```

8. (Optional) In priviliged EXEC mode, enter the **show ipv6 fabric-virtual-gateway interface ve** command to view the configured settings.

```
device# show ipv6 fabric-virtual-gateway interface ve
```

The following example shows how to configure an IPv6 Fabric-Virtual-Gateway on a VE interface.

```
device(config)# interface ve 3000
device(config-Ve-3000)# attach rbridge-id add 54-56
device(config-Ve-3000)# ipv6 fabric-virtual-gateway
device(config-ipv6-fabric-virtual-gw)# enable
device(config-ipv6-fabric-virtual-gw)# gateway-address 2001:1:0:1::1/64
device(config-ipv6-fabric-virtual-gw)# hold-time 30
device(config-ipv6-fabric-virtual-gw)# gratuitous-arp timer 15
device(config-ipv6-fabric-virtual-gw)# load-balancing-disable
```

The following example shows the output of the **show ipv6 fabric-virtual-gateway interface ve** command.

```
============Rbridge-id:54============
Interface        Admin    State       Gateway          ARP        Load       Threshold    Track
                 State                IP Address       Responder  Balancing  Priority     Priority
==============   =====    =======     =========        =========  =========  ==========   ========
Ve 3000          Enabled  Active      2001:1:0:1::1/64 Rbr-id 56  Enabled    unset        0
```

Configuring Fabric-Virtual-Gateway on an RBridge VE interface (IPv4)

To configure Fabric-Virtual-Gateway on a VE interface for an RBridge, perform the following steps.

NOTE
The ARP responder is automatically selected in a network. Enter the **clear ip fabric-virtual-gateway all** command in privileged EXEC mode to retrigger the election of a new ARP responder.

1. Enter the **rbridge-id** command in global configuration mode.

```
device(config)# rbridge-id 55
```

2. Enter the **interface ve** command.

```
device(config-rbridge-id-55)# interface ve 2000
```

3. Enter the **ip fabric-virtual-gateway** command to enable the IPv4 Fabric-Virtual-Gateway configuration.

```
device(config-rbridge-Ve-2000)# ip fabric-virtual-gateway
```

NOTE
If the Fabric-Virtual-Gateway configuration is disabled, enter the **enable** command to enable it.

4. (Optional) Enter the **disable** command to disable the Fabric-Virtual-Gateway configuration for a particular RBridge ID.

```
device(config-ip-fabric-virtual-gw)# disable
```

5. (Optional) Enter the **load-balancing threshold-priority** command to set the load balancing threshold value.

```
device(config-ip-fabric-virtual-gw)# load-balancing threshold-priority 150
```

6. (Optional) Enter the **track** followed by the track priority to track a physical interface, network, or next hop configuration.

NOTE
Only the local interfaces can be tracked.

```
device(config-ip-fabric-virtual-gw)# track interface fortygigabitethernet 55/0/51 priority 100
```

The following example shows how to configure IPv4 Fabric-Virtual-Gateway for an interface VE in an RBridge ID.

```
device(config)# rbridge-id 55
device(config-rbridge-id-55)# interface ve 2000
device(config-rbridge-Ve-2000)# ip fabric-virtual-gateway
device(config-ip-fabric-virtual-gw)# enable
device(config-ip-fabric-virtual-gw)# load-balancing threshold-priority 150
device(config-ip-fabric-virtual-gw)# track interface fortygigabitethernet 55/0/51 priority 100
device(config-ip-fabric-virtual-gw)# track network 192.128.2.0/24 priority 150
```

Configuring Fabric-Virtual-Gateway on an RBridge VE interface (IPv6)

To configure Fabric-Virtual-Gateway on a VE interface for an RBridge, perform the following steps.

NOTE
The ARP responder is automatically selected in a network. Enter the **clear ipv6 fabric-virtual-gateway all** command in privileged EXEC mode to retrigger the election of a new ARP responder.

1. Enter the **rbridge-id** command in global configuration mode.

```
device(config)# rbridge-id 58
```

2. Enter the **interface ve** command.

```
device(config-rbridge-id-58)# interface ve 2000
```

3. Enter the **ipv6 fabric-virtual-gateway** command to enable the IPv6 Fabric-Virtual-Gateway configuration.

```
device(config-rbridge-Ve-2000)# ipv6 fabric-virtual-gateway
```

NOTE
If the Fabric-Virtual-Gateway configuration is disabled, enter the **enable** command to enable it.

4. (Optional) Enter the **disable** command to disable the Fabric-Virtual-Gateway configuration for a particular RBridge ID.

```
device(config-ipv6-fabric-virtual-gw)# disable
```

5. (Optional) Enter the **load-balancing threshold-priority** command to set the load balancing threshold value.

```
device(config-ipv6-fabric-virtual-gw)# load-balancing threshold-priority 100
```

6. (Optional) Enter the **track** followed by the track priority to track a physical interface, network, or next hop configuration.

   NOTE
   Only the local interfaces can be tracked.

```
device(config-ipv6-fabric-virtual-gw)# track network 4001:1:0:1::/64 priority 150
```

The following example shows how to configure IPv6 Fabric-Virtual-Gateway for an interface VE in an RBridge ID.

```
device(config)# rbridge-id 58
device(config-rbridge-id-58)# interface ve 2000
device(config-rbridge-Ve-2000)# ipv6 fabric-virtual-gateway
device(config-ipv6-fabric-virtual-gw)# enable
device(config-ipv6-fabric-virtual-gw)# load-balancing threshold-priority 100
device(config-ipv6-fabric-virtual-gw)# track network 4001:1:0:1::/64 priority 150
```

Troubleshooting Fabric-Virtual-Gateway

Fabric-Virtual-Gateway sessions can conflict with VRRP, VRRP-E, or other configurations on an RBridge; for example, IP address configurations can conflict.

Because the scope of conflicts is limited to specific RBridges, a given session may not incur conflicts on some RBridges but does incur multiple conflicts on others. At a high level, conflicts are categorized as follows:

- Global-level conflicts
- Address-family-level conflicts
- Session-level conflicts

When conflicts are detected, the conflicted session is automatically disabled, to maintain appropriate functionality.

Global-level conflicts

A global-level conflict can arise on VDX 6740 series and VDX 6940 series platforms where only one or two unique gateway MAC addresses are allowed in the system. In such cases, when a conflict is detected none of the sessions belonging to the address family is allowed to become active. The summary output in such cases shows the reason for the conflict, and all of the sessions remain in the "Init" state, as shown in the following example.

```
device# show ip fabric-virtual-gateway
============Rbridge-id:3============
Total number of IPv4 Fabric Virtual Gateway sessions   : 3
Total number of sessions in Active state   : 0
Total number of sessions in InActive state : 0
Total number of sessions in Init state     : 3
Gateway MAC address: 02e0.5200.01ff
Configuration disabled due to reason(s): Gateway MAC address conflict

Interface  Admin     State        Gateway        ARP        Load       Threshold  Track
           State                   IP Address     Responder  Balancing  Priority   Priority
=========  =====     =====    ================== =========  =========  =========  ==========
Ve 100     Enabled   Init     8.3.1.100/24                  Enabled     unset      0
Ve 200     Enabled   Init     8.3.2.100/24                  Enabled     unset      0
Ve 300     Enabled   Init     8.3.3.100/24                  Enabled     unset      0
```

Address-family-level conflicts

When an address-family error status resulting from a conflict is not removed, even after the conflicting configuration is removed (for example, VRRP or VRRP-E is unconfigured from the RBridge), the user can execute a **clear** command, as in the IPv4 example below, to re-evaluate the conflict. If no conflicts are present, the error status is removed and all sessions are activated.

```
device# clear ip fabric-virtual-gateway all
```

Session-level conflicts

Session-level conflicts may arise as a result of gateway IP address/subnet conflicts with a gateway IP address/subnet in other Fabric-Virtual-Gateway sessions, virtual IP addresses of VRRP/VRRP-E sessions, or interface IP addresses on the same or other interfaces on an RBridge. There may also be other reasons for conflicts. When one or more conflicts are detected, the Fabric-Virtual-Gateway session is deactivated. The error status due to the conflict can be seen in the detailed output of the **show ip fabric-virtual-gateway** command for the VE interface, as follows:

```
device(config-ip-fabric-virtual-gw)# do show ip fabric-virtual-gateway int ve 2 detail

============Rbridge-id:56============
Interface: Ve 2;  Ifindex: 1207959554
  Admin Status: Enabled
  Description :
  Address family: IPV4    State: Init
  Error(s):
         Conflicting protocol configuration found on interface
         Gateway IP address conflicts with VRRP/VRRP-E session
  ARP responder Rbridge-id:
  Gateway IP: 1.2.0.10/24
  Gateway MAC Address: 02e0.5200.01ff
  Load balancing configuration: Enabled
  Load balancing current status: Disabled
  Load balancing threshold priority: unset
  Gratuitous ARP Timer: Disabled
  Hold time: 0 sec  (default: 0 sec)
    Total no. of state changes: 0
    Gratuitous ARP Sent: 0
  Last state change: 0d.0h.1m.52s ago
  Track Priority: 0
```

Static Anycast Gateway

Overview of static anycast gateway

Static anycast gateway enables you to configure identical IP/MAC gateway addresses to virtual Ethernet (VE) interfaces on leaf switches in an IP Fabric, increasing routing efficiency.

Static anycast gateway provides seamless virtual machine (VM) mobility across all of the leaf (ToR) switches . Even if hosts move among leaf switches, there is no need to reconfigure the default gateway. In this way, forwarding behavior is optimized.

Considerations and limitations for static anycast gateway

There are considerations and limitations that you need to be aware of when implementing static anycast gateway.

Static anycast gateway is supported only in an IP Fabric, and BGP EVPN must be enabled. In addition, when this feature is enabled in an IP Fabric, to prevent ARP/ND broadcast across the network, the user should enable ARP/ND suppression on the respective VLANs.

Static anycast gateway is not supported along with Fabric-Virtual-Gateway (FVG).

The following tables summarize support for VRRP and VRRP-E with static anycast gateway on the supported platforms.

TABLE 21 Support for static anycast gateway with VRRP and VRRP-E (VDX 6740 series)

Static anycast gateway MAC address	VRRP (IPv4 or IPv6)	VRRP-E (IPv4 or IPv6)
Default	No	Yes
Nondefault	No	No

TABLE 22 Support for static anycast gateway with VRRP and VRRP-E (VDX 6940 series)

Static anycast gateway MAC address	VRRP (IPv4 or IPv6)	VRRP-E (IPv4 or IPv6)
Default	Yes	Yes
Nondefault	Yes	No

Configuring MAC static anycast gateway addresses

To implement static anycast gateway, you need to specify gateway MAC addresses for IPv4 and IPv6 traffic on each RBridge.

NOTE
Depending on your needs, you can specify a MAC address for IPv4 traffic, for IPv6 traffic, or for both.

1. Enter **configure terminal** to access global configuration mode.

```
device# configure terminal
```

2. Enter the **rbridge-id** command to access RBridge ID configuration mode.

```
device(config)# rbridge-id 1
```

3. To specify a static anycast gateway MAC address for IPv4 traffic, enter the **ip anycast-gateway-mac** command, with one of the following options:
   - To specify the default static anycast gateway MAC address for IPv4 traffic, enter the following command:

```
device(config-rbridge-id-1)# ip anycast-gateway-mac default-mac
```

     NOTE
     "Default" means that an automatically generated MAC address is used.

   - To specify a nondefault static anycast-gateway MAC address for IPv4 traffic, enter a command such as the following:

```
device(config-rbridge-id-1)# ip anycast-gateway-mac 2222.2244.4444
```

4. To specify a static anycast gateway MAC address for IPv6 traffic, enter the **ipv6 anycast-gateway-mac** command, with one of the following options:
   - To specify the default static anycast gateway MAC address for IPv6 traffic, enter the following command:

```
device(config-rbridge-id-1)# ipv6 anycast-gateway-mac default-mac
```

   - To specify a nondefault static anycast gateway MAC address for IPv6 traffic, enter a command such as the following:

```
device(config-rbridge-id-1)# ipv6 anycast-gateway-mac 2222.2266.6666
```

     NOTE
     The first three bytes (six digits) of a nondefault IPv4 static anycast gateway MAC address must be identical with the first three bytes of a corresponding IPv6 static anycast gateway MAC address.

Configuring IP static anycast gateway addresses

To implement static anycast gateway, you must specify IPv4 and IPv6 static anycast gateway addresses on a virtual Ethernet (VE) interface.

NOTE
Depending on your needs, you can specify an IPv4 address, an IPv6 address, or both.

1. Enter **configure terminal** to access global configuration mode.

```
device# configure terminal
```

2. Enter the **rbridge-id** command to access RBridge ID configuration mode.

```
device(config)# rbridge-id 1
```

3. Enter the **interface ve** command to access VE (RBridge) interface configuration mode.

```
device(config-rbridge-id-1)# interface ve 10
```

4. To specify an IPv4 static anycast gateway address, enter the **ip anycast-address** command.

```
device(config-rbridge-Ve-1)# ip anycast-address 2.2.2.2/24
```

5. To specify an IPv6 static anycast gateway address, enter the **ipv6 anycast-address** command.

```
device(config-rbridge-Ve-1)# ipv6 anycast-address 1234:10::10:100/64
```

Show commands for static anycast gateway

Use the following **show** commands to verify configurations of static anycast gateway.

TABLE 23 Show commands for static anycast gateway

Command	Description
show ip anycast-gateway	Displays IPv4 static anycast gateway details for all virtual Ethernet (VE) interfaces or for a specified VE interface. You can also filter by RBridge and by VRF.
show ipv6 anycast-gateway	Displays IPv6 static anycast gateway details for all VE interfaces or for a specified VE interface. You can also filter by RBridge and by VRF.

Traceroute for Overlay Tunnels

Traceroute for overlay tunnels overview

The tunnel traceroute feature provides underlay hop information and overlay reachability information in an overlay tunnel, for enhanced visibility of packet traversal between network virtualization edge (NVE) nodes in a Clos IP Fabrics network.

The generic traceroute mechanism, as exemplified by the **l2traceroute** command, does not take Equal Cost Multipath (ECMP) into account, and so is not helpful in cases where traffic on a particular ECMP path is blackholed. This feature, supported on theVDX 6740, VDX 6940, and VDX 8770 series platforms, uses the **tunnel-traceroute** command to address this problem.

This feature supports both IPv4 and IPv6. There is no difference in the mechanism for IPv6 flows, because underlay routing is still IPv4-based. The traceroute mechanism parses the IPv6 flow parameters entered by the user and initiates a traceroute session. It constructs the VXLAN-encapsulated packet with the inner IPv6 header to obtain the ECMP-hashing-based IPv6 flow parameters.

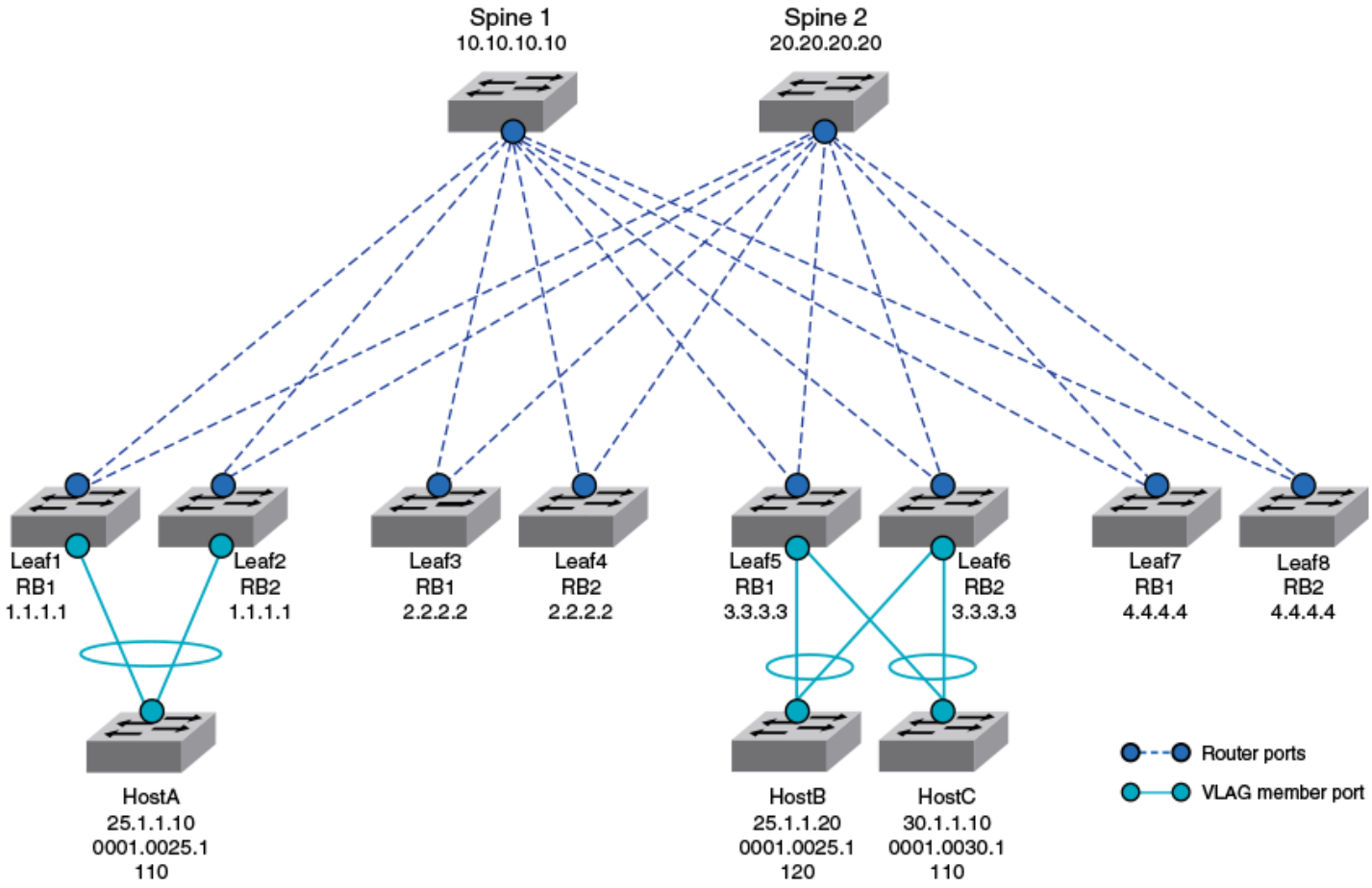
NOTE
The VDX 8770 series can be a spine node in an IP Fabrics network.

The generic traceroute mechanism sends probe packets by incrementing the Time To Live (TTL) for each hop. Transit nodes in the underlay path respond to TTL expiration with an ICMP error message. An egress NVE device responds with either an ICMP reply with reason code "Port unreachable" or a UDP response with additional information regarding tunnel termination on the node.

With the **tunnel-traceroute** command, if a request packet is tunnel terminated, the egress NVE node process the packet and generates a UDP response message with details regarding tunnel termination and VNI-to-VLAN mappings at the node. If the packet is not tunnel terminated, an ICMP error message is sent to the ingress NVE.

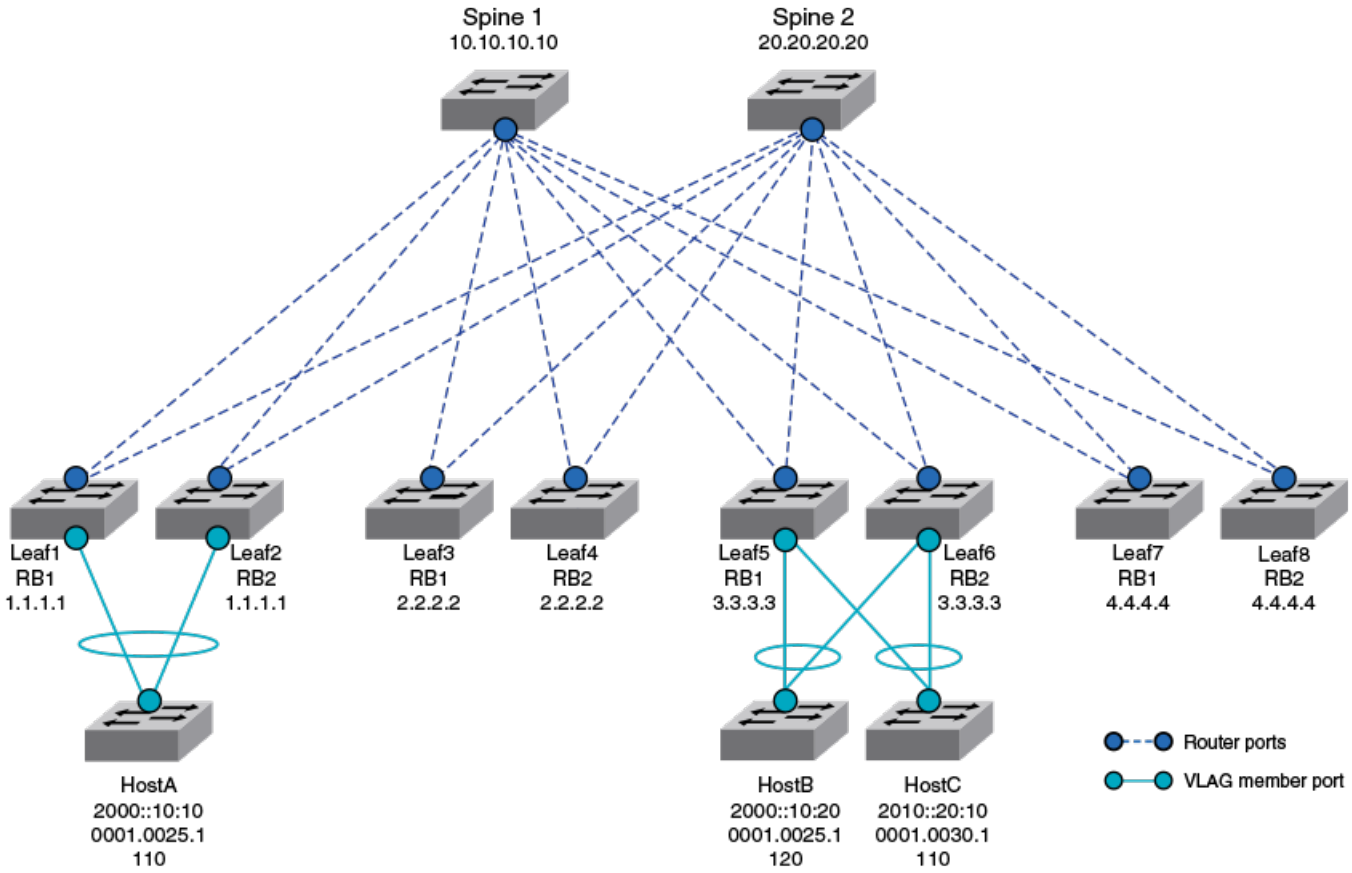
The following is an example Clos IP Fabrics IPv4 topology for a single data-center pod. Host A and Host B are in same subnets but are connected to different leaf nodes (dual-homed to Leaf 1 and Leaf 2 and Leaf 5 and Leaf 6, respectively) and Host C is in different subnet and is connected to Leaf 3 and Leaf 4. Assume that traffic forwarding between Host A and Host B takes an ECMP path through Spine 1, and traffic forwarding between Host A and Host C takes another ECMP path through Spine 2 to reach Leaf 3 from Leaf 1. Assume also that Host A is dual-homed to Leaf 1 and Leaf 2 through VLAG 10.

FIGURE 60 Example Clos IP Fabrics IPv4 topology for a single data-center pod



The following is an example Clos IP Fabrics IPv6 topology for a single data-center pod. Host A and Host B are in same subnets but are connected to different leaf nodes (dual-homed to Leaf 1 and Leaf 2 and Leaf 5 and Leaf 6, respectively). Host C is in a different subnet and is connected to Leaf 3 and Leaf 4. Assume that traffic forwarding between Host A and Host B takes an ECMP path through Spine 1, and forwarding between Host A and Host C takes an ECMP path through Spine 2 to reach Leaf 3 from Leaf 1. Assume that Host A is dual-homed to Leaf 1 and Leaf 2 through VLAG 10

FIGURE 61 Example Clos IP Fabrics IPv6 topology for a single data-center pod



Before you can execute the **tunnel-traceroute** command, you must initialize an ASIC simulator and update registers and memory with current values in hardware. This is done by means of the **debug asic-simulation-model load** command. You can refresh the status of memory and registers by using the **debug asic-simulation model refresh** command, and you can see the status of this operation by using the **show debug asic-simulation model** command. Refer to the *Extreme Network OS Command Reference* for details.

Using traceroute for overlay tunnels

This feature provides underlay hop information and overlay reachability information in a overlay tunnel, for enhanced visibility of packet traversal between network virtualization edge (NVE) nodes in a Clos IP Fabrics network. Both IPv4 and IPv6 are supported.

The following IPv4 and IPv6 examples illustrate the use of the **tunnel-traceroute** command for the example topology.

The following IPv4 example returns a successful tunnel termination at the egress network virtualization edge (NVE).

```
device# tunnel-traceroute sa 0001.0025.1110 da 0001.0025.1120 ipv4 sip 25.1.1.10 DIP 25.1.1.20 vlan 100
   protocol udp l4-src-port 1000 4-dest-port 2000 interface port-channel 10
Hop count     next hop         RTT1         RTT2         RTT3
1        10.10.10.10        0.345 ms      0.439 ms      0.418 ms
2.        3.3.3.3            0.432 ms      0.380ms    0.402 ms
        Packet is tunnel terminated at NVE 3.3.3.3 and rbridge id 1
VNI 1000 -> VLAN 100
```

The following IPv4 example returns an unsuccessful tunnel termination at the NVE.

```
device# tunnel-traceroute sa 0001.0025.1110 da 0001.0030.1110 ipv4 sip 25.1.1.10 dip 30.1.1.10 vlan 100
Hop count     next hop         RTT1         RTT2         RTT3
1        20.20.20.20        0.395 ms      0.367 ms      0.411 ms
2.        3.3.3.3            0.419 ms      0.388ms    0.456 ms
        Packet is not tunnel terminated at NVE 3.3.3.3 and rbridge id 1
```

NOTE

The above output also occurs when nodes are configured not to respond to TTL expiry errors but Network OS switches are configured to respond to those errors with ICMP replay messages.

The following IPv4 example can be used to isolate a fault in an IP Fabric with all Network OS switches.

```
 device# tunnel-traceroute sa 0001.0025.1110 da 0001.0030.1110 ipv4 sip 25.1.1.10 dip 30.1.1.10 vlan 100
  protocol udp l4-src-port 1000 4-dest-port 2000 interface Te 1/0/20
Hop count     next hop         RTT1         RTT2         RTT3
1        10.10.10.10        0.395 ms      0.367 ms      0.411 ms
2.        *            *          *        *
3.         *            *        *        *
……..
…….
30.        *            *        *        *
```

The following IPv6 examples return a successful tunnel termination at the egress network virtualization edge (NVE).

```
device# tunnel-traceroute SA 0001.0025.1110 DA 0001.0025.1120 IPV6 SIP 2000::10:10  DIP 2000::10:20 VLAN
100 interface port-channel 10 protocol UDP L4-Src-Port 1000 4-dest-port 2000
Hop count     next hop         RTT1         RTT2         RTT3
1        10.10.10.10        0.345 ms      0.439 ms      0.418 ms
2.        3.3.3.3            0.432 ms      0.380ms    0.402 ms
        Packet is tunnel terminated at NVE 3.3.3.3 and rbridge id 1
VNI 1000 -> VLAN 100


device# tunnel-traceroute SA 0001.0025.1110 DA 0001.0030.1110 IPV6 SIP 2000::10:10 DIP 2010::20:10 VLAN 100
interface port-channel 10protocol UDP L4-src-port 1000 4-dest-port 2000
Hop count     next hop         RTT1         RTT2         RTT3
1        20.20.20.20        0.395 ms      0.367 ms      0.411 ms
2.        3.3.3.3            0.419 ms      0.388ms    0.456 ms
        Packet is tunnel terminated at NVE 3.3.3.3 and rbridge id 1
VNI 2000 -> VLAN 200
```