Extreme SLX-OS
Layer 3 Routing Configuration Guide, 20.1.1

Supporting ExtremeRouting and ExtremeSwitching
SLX 9640, SLX 9540, SLX 9150, and SLX 9250

Table of Contents

Preface

This section discusses the conventions used in this guide, ways to provide feedback, additional help, and other Extreme Networks® publications.

Text Conventions

Unless otherwise noted, information in this document applies to all supported environments for the products in question. Exceptions, like command keywords associated with a specific software version, are identified in the text.

When a feature, function, or operation pertains to a specific hardware product, the product name is used. When features, functions, and operations are the same across an entire product family, such as ExtremeSwitching switches or SLX routers, the product is referred to as *the switch* or *the router*.

Table 1: Notes and warnings

Icon	Notice type	Alerts you to...
	Tip	Helpful tips and notices for using the product.
	Note	Useful information or instructions.
	Important	Important features or instructions.

Table 1: Notes and warnings (continued)

Icon	Notice type	Alerts you to...
⚠️	Caution	Risk of personal injury, system damage, or loss of data.
⚠️	Warning	Risk of severe personal injury.

Table 2: Text

Convention	Description
`screen displays`	This typeface indicates command syntax, or represents information as it appears on the screen.
The words *enter* and *type*	When you see the word *enter* in this guide, you must type something, and then press the Return or Enter key. Do not press the Return or Enter key when an instruction simply says *type*.
Key names	Key names are written in boldface, for example **Ctrl** or **Esc**. If you must press two or more keys simultaneously, the key names are linked with a plus sign (+). Example: Press **Ctrl**+**Alt**+**Del**
Words in italicized type	Italics emphasize a point or denote new terms at the place where they are defined in the text. Italics are also used when referring to publication titles.
NEW!	This symbol identifies new content. In a PDF, this is searchable text.

Table 3: Command syntax

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, such as 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.

Documentation and Training

Find Extreme Networks product information at the following locations:

Current Product Documentation
Archived Documentation (for earlier versions and legacy products)
Release Notes
Hardware/software compatibility matrices for Campus and Edge products
Supported transceivers and cables for Data Center products
Other resources, like white papers, data sheets, and case studies

Extreme Networks offers product training courses, both online and in person, as well as specialized certifications. For details, visit www.extremenetworks.com/education/.

Getting Help

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

Extreme Portal

Search the GTAC (Global Technical Assistance Center) knowledge base; manage support cases and service contracts; download software; and obtain product licensing, training, and certifications.

The Hub

A forum for Extreme Networks customers to connect with one another, answer questions, and share ideas and feedback. This community is monitored by Extreme Networks employees, but is not intended to replace specific guidance from GTAC.

Call GTAC

For immediate support: (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

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

- Your Extreme Networks service contract number, or serial numbers for all involved Extreme Networks products
- A description of the failure
- A description of any actions 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

Subscribe to Service Notifications

You can subscribe to email notifications for product and software release announcements, Vulnerability Notices, and Service Notifications.

1. Go to www.extremenetworks.com/support/service-notification-form.
2. Complete the form (all fields are required).
3. Select the products for which you would like to receive notifications.

> **Note**
> You can modify your product selections or unsubscribe at any time.

4. Select **Submit**.

Providing Feedback

The Information Development team at Extreme Networks has made every effort to ensure the accuracy and completeness of this document. We are always striving to improve our documentation and help you work better, so we want to hear from you. We welcome all feedback, but we especially want to know about:

- Content errors, or confusing or conflicting information.
- Improvements that would help you find relevant information in the document.
- Broken links or usability issues.

If you would like to provide feedback, you can do so in three ways:

- In a web browser, select the feedback icon and complete the online feedback form.
- Access the feedback form at https://www.extremenetworks.com/documentation-feedback/.
- Email us at documentation@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.

About This Document

Supported Hardware

For instances in which a topic or part of a topic applies to some devices but not to others, the topic specifically identifies the devices.

SLX-OS 20.1.1 supports the following hardware platforms.

- Devices based on the Broadcom XGS® chipset family:
  - ExtremeSwitching SLX 9250
  - ExtremeSwitching SLX 9150
- Devices based on the Broadcom DNX® chipset family:
  - ExtremeRouting SLX 9640
  - ExtremeSwitching SLX 9540

> **Note**
> Although many software and hardware configurations are tested and supported for this release, documenting all possible configurations and scenarios is beyond the scope of this document.

For information about other releases, see the documentation for those releases.

Address Resolution Protocol

ARP Overview

The Address Resolution Protocol (ARP) maps IPv4 network addresses to MAC hardware addresses.

When forwarding traffic, a device needs to know the destination MAC address because each IP packet is encapsulated in an Ethernet frame. The MAC address is needed for the packet's final destination and for a next hop toward the destination.

A device first searches its ARP cache and a match for the IP address supplies the corresponding MAC address. Otherwise, the device broadcasts an ARP request. Network devices receive the requests, and the host with a matching IP address sends an ARP reply that includes its MAC address.

After the device receives a matching ARP reply, the following events occur.

- The packet is sent toward its destination.
- The IP address/MAC address pair is added to the ARP cache as a dynamic ARP entry.

> **Note**
> Neighbor Discovery is the technology by which a device gets the MAC address for IPv6. For more information, see IPv6 Neighbor Discovery on page 57.

Change the ARP Aging Timeout for an Interface

The ARP aging timeout for an interface overrides the global aging timeout of 25 minutes.

About This Task

An aging timer is triggered when a dynamic entry is added to the ARP cache and is reset if an ARP reply is received. The aging timer ensures that the ARP cache does not retain invalid learned entries. An entry can become invalid when the device with the MAC address of the entry is no longer on the network.

Procedure

1. Access global configuration mode.

   ```
   device# configure terminal
   ```

2. Access the interface on which you are changing the aging timeout.

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

3. Specify the new timeout value.

   ```
   device(conf-if-eth-0/1)# ip arp-aging-timeout 100
   ```

   Valid values range from 0 through 240 minutes.

Example

The following example summarizes the commands in this procedure.

```
device# configure terminal
device(config)# interface ethernet 0/1
device(conf-if-eth-0/1)# ip arp-aging-timeout 100
```

Enable ARP Learning

ARP learning decreases the time needed to populate the ARP cache.

About This Task

Use this procedure to enable ARP learning locally and from all ARP requests.

Procedure

1. Access global configuration mode.

   ```
   device# configure terminal
   ```

2. Access VE configuration mode.

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

3. Enable fabric learning.

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

Example

The following example summarizes the commands in this procedure.

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

Create a Static ARP Entry for an Interface

You can create a static ARP entry for a device that is not yet connected to the network or to prevent an entry from aging out.

About This Task
You can define a static ARP entry for a physical interface or for a VE. The examples in this topic reference a physical interface.

Procedure

1. Access global configuration mode.

   ```
   device# configure terminal
   ```

2. Create the static ARP entry.

   ```
   device(config)# arp 10.53.4.2 1245.7654.2348 interface ethernet 1/2
   ```

   The example creates a static ARP entry for IP address 10.53.4.2 and associates it with MAC address 1245.7654.2348 and physical port 1/2.

Example
The following example summarizes the commands in this procedure.

```
device# configure terminal
device(config)# arp 10.53.4.2 1245.7654.2348 interface ethernet 1/2
```

Create a Static ARP Entry for a VRF Instance

You can create a static ARP entry for a device that is not yet connected to the network or to prevent an entry from aging out.

Procedure

1. Access global configuration mode.

   ```
   device# configure terminal
   ```

2. Access VRF configuration mode.

   ```
   device(config)# vrf test
   ```

3. Specify the address family.

   ```
   device(config-vrf-test)# address-family ipv4 unicast
   ```

4. Create the static ARP entry.

   ```
   device(vrf-test-ipv4-unicast)# arp 10.6.6.7 0001.0001.0001 interface ethernet 0/1
   ```

   This example creates a static ARP entry for IP address 10.6.6.7 and associates it with MAC address 0001.0001.0001 and Ethernet interface 0/1.

Example
The following example summarizes the commands in this procedure.

```
device# configure terminal
device(config)# vrf test
device(config-vrf-test)# address-family ipv4 unicast
device(vrf-test-ipv4-unicast)# arp 10.6.6.7 0001.0001.0001 interface ethernet 0/1
```

Proxy ARP

With the Proxy ARP functionality enabled, a device answers ARP requests from devices in one network on behalf of devices in another network.

Because Address Resolution Protocol (ARP) requests are MAC-layer broadcasts, they reach only the devices that are directly connected to the sender of the ARP request. ARP requests do not cross routers. However, when Proxy ARP is enabled on a device that is connected to two subnets, the device can respond to an ARP request from the other subnet.

For example:

* The device is connected to the 10.10.10.0/24 and 10.20.20.0/24 subnets.
* The device can respond to an ARP request from 10.10.10.69 for the MAC address of the device with the IP address 10.20.20.69.
* In standard ARP, a request from a device in the 10.10.10.0/24 subnet cannot reach a device in the 10.20.20.0 subnet.

The ARP reply contains the device's MAC address instead of the MAC address of the target host. In this transaction, the traffic sent to the target host is forwarded through Layer 3 rather than being switched through Layer 2.

> **Note**
> Under some Layer 2 configurations, such as an uplink switch or private VLAN, broadcast packets are not flooded to every port in a VLAN. In these configurations, an ARP request from one host may not reach another host. When Proxy ARP is enabled locally on a port, the device is directed to reply on behalf of a target host if it exists.

Enable Proxy ARP on an Interface

You can enable Proxy ARP on a physical or VE interface.

About This Task
Proxy ARP is disabled by default.

Procedure

1. Access global configuration mode.

   ```
   device# configure terminal
   ```

2. Access the interface on which you are enabling Proxy ARP.

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

3. Enable Proxy ARP.

   ```
   device(conf-if-eth-0/1)# ip proxy-arp
   ```

Example
The following example summarizes the commands in this procedure.

```
device# configure terminal
device(config)# interface ethernet 0/1
device(conf-if-eth-0/1)# ip proxy-arp
```

NEW! ARP Poisoning

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

An Address Resolution Protocol (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).

Two features protect against ARP poisoning.

- Dynamic ARP Inspection (DAI) on page 28
- ARP Guard on page 31

Table 4: Comparison of ARP Guard and DAI

Aspect	DAI	ARP Guard
Flow-based	No. Applies to all VLAN ARP packets.	Flow-based, which can prevent high CPU load.
Per port	No. Applies to all VLAN ports.	Applied per port or VPLS end-point.
Rate-limiting	No rate-limiting option.	Rate limiting is supported.
TCAM load	Low TCAM load.	Medium TCAM load.

Create an ARP Access Control List

You can create an ARP ACL to use with Dynamic ARP Inspection (DAI) and ARP Guard.

Procedure

1. Access global configuration mode.

```
device# configure terminal
```

2. Create the ACL.

```
device(config)# arp access-list ARP_ACL_01
```

This example creates an ACL named ARP_ACL_01.

3. For each ACL rule, specify the IP address and MAC address pairs that are allowed access.

```
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
```

Example

The following example summarizes the commands in this procedure.

```
device# configure terminal
device(config)# arp access-list ARP_ACL_01
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
```

Dynamic ARP Inspection (DAI)

The Dynamic ARP Inspection (DAI) security feature validates Address Resolution Protocol (ARP) packets in a subnet, and discards packets with invalid IP address and MAC address bindings.

> **Note**
> DAI is supported only in non-DHCP environments.

On VLANs, DAI can examine incoming ARP packets. DAI discards packets with invalid IP address and MAC address bindings, guarding against ARP poisoning. Only valid ARP requests and responses are relayed. You specify valid, static IP address and MAC address bindings in the **permit** statements of ARP ACLs.

You decide which ports to define as trusted or untrusted. ARP packets on trusted ports bypass all DAI validations and are forwarded as required. DAI examines ARP packets only on untrusted ports.

DAI monitors untrusted ports as follows.

- Intercepts ARP requests and responses.
- Compares the IP address and MAC address bindings with the **permit** statements in the ACL applied to the VLAN.

- Drops invalid packets.
- Forwards valid packets to the appropriate destination.

Table 5: DAI on trusted and untrusted ports

VLAN setting	Port setting	Action
DAI disabled	Trusted or untrusted	All incoming ARP packets are hardware-forwarded.
DAI enabled	Trusted	All incoming ARP packets are trapped to the CPU and then software-forwarded.
DAI enabled	Untrusted	All incoming ARP packets are trapped to the CPU. Following DAI, the packets are software-forwarded or dropped.

Guidelines for Implementing ACLs for DAI

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

Follow these guidelines when implementing Address Resolution Protocol (ARP) ACLs for DAI.

- DAI is available on the following Layer 2 VLANs.
  - 802.1Q VLANs
  - VE interfaces under virtual routing and forwarding (VRF). Both default and non-default VRFs are supported.
- DAI is not supported for management 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 MLAGs) (in switchport mode)

Apply an ARP ACL to a VLAN

For DAI, use this procedure to apply an ARP ACL to a VLAN.

About This Task
The most recently applied ACL replaces an existing ACL. You do not need to remove an existing ACL.

Procedure

1. Access global configuration mode.

```
device# configure terminal
```

2. Access VLAN configuration mode.

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

3. Specify the ACL that you want to apply.

   For more information, see Create an ARP Access Control List on page 27.

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

4. Return to global configuration mode to, for example, define trusted and untrusted interfaces and to enable DAI.

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

Example

The following example summarizes the commands in this procedure.

```
device# configure terminal
device(config)# vlan 200
device(config-vlan-200)# ip arp inspection filter ARP_ACL_01
device(config-vlan-200)# exit
```

Define Trusted and Untrusted Interfaces for DAI

You can define the trusted and untrusted interfaces under Dynamic ARP Inspection (DAI).

About This Task

An interface is untrusted by default.

Procedure

1. Access global configuration mode.

   ```
   device# configure terminal
   ```

2. Access interface configuration mode.

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

3. To define an interface as trusted, run the **ip arp inspection trust** command.

   ```
   device(conf-if-eth-2/3)# ip arp inspection trust
   ```

4. To define a trusted interface as untrusted, run the **no ip arp inspection trust** command.

   ```
   device(conf-if-eth-2/3)# no ip arp inspection trust
   ```

Example

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
```

Example

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

```
device# configure terminal
device(config)# interface port-channel 200
device(config-Port-channel-200)# no ip arp inspection trust
```

Enable DAI on a VLAN

You can enable Dynamic ARP Inspection (DAI) on a VLAN.

Procedure

1. Access global configuration mode.

    ```
    device# configure terminal
    ```

2. Access configuration mode on the VLAN for which you are enabling DAI.

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

3. Enable DAI.

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

Example

The following example summarizes the commands in this procedure.

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

DAI Show and Clear Commands

You can display and delete Dynamic ARP Inspection (DAI) information.

Table 6: DAI show and clear commands

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

ARP Guard

ARP Guard is an alternative to Dynamic ARP Inspection (DAI) for protection against ARP poisoning.

> **Note**
> ARP Guard is supported only on devices based on the DNX chipset family. For a list of such devices, see Supported Hardware on page 22.

Internet exchange points (IXPs) have a flat Layer 2 topology to provide any-to-any connectivity among BGP routers from connected ISPs, CSPs, and enterprises. As an IP host, each BGP peering router uses Address Resolution Protocol (ARP) to determine the MAC address of its BGP peers.

Because ARP is not a secure protocol, any BGP router can reply to the ARP request for any IP address. And any BGP router can generate gratuitous ARP to claim ownership of any IP address in the router. Valid traffic can be sent to the wrong destination in the following scenarios.

- The IP address of a BGP router (connected to the IXP) is misconfigured.
- The network administrator unknowingly turns on the ARP Proxy feature on the interface that faces the IXP.

ARP Guard, like DAI, is effective against the various methods of ARP poisoning. For more information, see ARP Poisoning on page 27.

The ARP Guard feature uses a set of ACL-like commands to build a table of allowed IP addresses on the link. As a result, when an ARP reply—either due to gratuitous ARP or in response to a normal ARP request—is received on a port facing the BGP router, the reply is compared to the table of allowed IP addresses. ARP packets that do not match the entries are dropped. Matching ARP packets are forwarded.

For more information about the ACL, see Create an ARP Access Control List on page 27.

ARP Guard Configuration Guidelines

Implementing ARP Guard requires creating ARP ACLs, then applying and enabling them on interfaces and port-channels.

Follow these guidelines when implementing ARP Guard.

- ARP Guard is supported on physical interfaces and port-channels.
- ARP Guard is not supported on Layer 3 interfaces.
- ARP Guard requires that the **switchport** command be implemented on the interface.
- If ARP Guard is enabled on a port-channel and a member port is removed, ARP Guard properties are retained on the removed port.
- On an interface where ARP Guard is enabled, you must limit the rate of ARP traffic by a Layer 2 ACL. For more information about the ACL, see Create an ARP Access Control List on page 27.

The following table indicates which hardware profiles support the various types of ARP Guard implementation. For more information, see the **show hardware profile** command in *Extreme SLX-OS Command Reference*.

Table 7: Profile Support for ARP Guard

Profile	Basic ARP Guard	ARP Guard with Rate Limiting	ARP Guard on VPLS Endpoint	ARP Guard on VPLS End-point with Rate Limiting
default	Yes	Yes	Yes	Yes
vxlan-visibility	Yes	No	No	No

Apply an ARP ACL to an Interface or Port-channel

For ARP Guard, use this procedure to apply an ARP ACL to an interface or port-channel.

About This Task

The most recently applied ACL replaces an existing ACL. You do not need to remove an existing ACL.

Procedure

1. Access global configuration mode.

```
device# configure terminal
```

2. Access interface configuration mode.

```
device(conf)# interface ethernet 1/2
```

3. Access Layer 2 mode.

```
device(conf-if-eth-1/2)# switchport
```

4. Specify the ACL that you want to apply.

   For more information, see Create an ARP Access Control List on page 27.

```
device(conf-if-eth-1/2)# ip arp inspection filter ARP_ACL_01
```

5. Return to global configuration mode.

```
device(conf-if-eth-1/2)# exit
device(conf)#
```

Example

The following example summarizes the commands in this procedure.

```
device# configure terminal
device(conf)# interface ethernet 1/2
device(conf-if-eth-1/2)# switchport
device(conf-if-eth-1/2)# ip arp inspection filter ARP_ACL_01
device(conf-if-eth-1/2)# exit
```

Enable ARP Guard on an Interface or Port-channel

You can enable ARP Guard on a physical interface or a port-channel.

Procedure

1. Access global configuration mode.

```
device# configure terminal
```

2. Create an ACL to enable ARP Guard.

```
device(config)# mac access-list extended arp_guard_enable_1
```

   This example creates an ACL named arp_guard_enable_1.

3. In the ACL, create the rules that you want to implement.

```
device(conf-macl-ext)# permit host 0014.2211.1111 any vlan 100 arp arp-guard
device(conf-macl-ext)# permit host 0014.2211.1112 any vlan 101 arp arp-guard
device(conf-macl-ext)# deny any any arp
device(conf-macl-ext)# permit any any
```

4. Access interface configuration mode.

```
device(conf)# interface ethernet 1/2
```

5. Access Layer 2 mode.

```
device(conf-if-eth-1/2)# switchport
```

6. Specify the ACL and the **in** direction.

```
device(conf-if-eth-1/2)# mac access-group arp_guard_enable_1 in
```

Example

The following example summarizes the commands in this procedure.

```
device# configure terminal
device(config)# mac access-list extended arp_guard_enable_1
device(conf-macl-ext)# permit host 0014.2211.1111 any vlan 100 arp arp-guard
device(conf-macl-ext)# permit host 0014.2211.1112 any vlan 101 arp arp-guard
device(conf-macl-ext)# deny any any arp
device(conf-macl-ext)# permit any any
device(conf)# interface ethernet 1/2
device(conf-if-eth-1/2)# switchport
device(conf-if-eth-1/2)# mac access-group arp_guard_enable_1 in
```

Implement Rate Limiting for ARP Guard

You can implement ACL-based rate limiting for ARP Guard on a physical interface or a port-channel.

About This Task

For more information about ACL-based rate limiting, see the *Extreme SLX-OS QoS and Traffic Management Configuration Guide*.

Procedure

1. Access global configuration mode.

```
device# configure terminal
```

2. Create an ACL.

```
device(config)# mac access-list extended rate-limit-acl_1
```

This example creates an ACL named rate-limit-acl_1.

3. Create the ACL rules that you want to implement.

```
device(conf-macl-ext)# permit host 0014.2211.1111 any vlan 100 arp arp-guard
device(conf-macl-ext)# permit host 0014.2211.1112 any vlan 101 arp arp-guard
device(conf-macl-ext)# deny any any arp
device(conf-macl-ext)# permit any any
```

4. Create a class map.

```
device(config)# class-map arp-guard-class
```

5. Associate the class map with the ACL.

```
device(config-classmap)# match access-group rate-limit-acl_1
```

6. Return to global configuration mode.

```
device(config-classmap)# exit
```

7. Create a policy map.

```
device(config)# policy-map arp-guard-pmap
```

8. Associate the class map with the policy map.

```
device(config-policymap)# class arp-guard-class
```

9. Specify class police parameters.

```
device(config-policymap-class)# police cir 100000
```

10. Return to global configuration mode.

```
device(config-policymap-class-police)# end
```

11. Enter configuration mode for the relevant interface.

```
device# configure terminal
device(config)# interface ethernet 1/2
```

12. Bind the policy map to the interface.

```
device(conf-if-eth-1/2)# service-policy in arp-guard-pmap
```

Example

The following example summarizes the commands in this procedure.

```
device# configure terminal
device(config)# mac access-list extended rate-limit-acl_1
device(conf-macl-ext)# permit host 0014.2211.1111 any vlan 100 arp arp-guard
device(conf-macl-ext)# permit host 0014.2211.1112 any vlan 101 arp arp-guard
device(conf-macl-ext)# deny any any arp
device(conf-macl-ext)# permit any any
device(config)# class-map arp-guard-class
device(config-classmap)# match access-group rate-limit-acl_1
device(config-classmap)# exit
device(config)# policy-map arp-guard-pmap
device(config-policymap)# class arp-guard-class
device(config-policymap-class)# police cir 100000
device(config-policymap-class-police)# end
device# configure terminal
device(config)# interface ethernet 1/2
device(conf-if-eth-1/2)# service-policy in arp-guard-pmap
```

ARP Suppression

In a data center fabric, you can suppress Address Resolution Protocol to help reduce ARP control traffic.

> **Note**
>
> ARP suppression is supported on VLANs and bridge domains.

By default, ARP is not suppressed, which 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 cache.
2. The device broadcasts an ARP request throughout the IP fabric.

When ARP is suppressed, 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 cache.
2. The device broadcasts an ARP 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 only to the device.

- If there is no match, the leaf control plane broadcasts the request to the entire IP fabric.

Tip
When the static anycast gateway feature is enabled in an IP fabric, suppress ARP on the respective VLANs or BDs to prevent ARP broadcast across the network.

Suppress ARP on a VLAN

You can suppress Address Resolution Protocol (ARP) on a VLAN.

Procedure

1. Access global configuration mode.

   ```
   device# configure terminal
   ```

2. Access VLAN configuration mode.

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

3. Suppress ARP.

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

Example
The following example summarizes the commands in this procedure.

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

Suppress ARP on a Bridge Domain

You can suppress Address Resolution Protocol (ARP) on a bridge domain.

Procedure

1. Access global configuration mode.

   ```
   device# configure terminal
   ```

2. Access bridge-domain configuration mode.

   ```
   device(config)# bridge-domain 2
   ```

3. Suppress ARP.

   ```
   device(config-bridge-domain-2)# suppress-arp
   ```

Example
The following example summarizes the commands in this procedure.

```
device# configure terminal
device(config)# bridge-domain 2
device(config-bridge-domain-2)# suppress-arp
```

ARP Suppression Show and Clear Commands

You can display and delete ARP suppression information.

Table 8: ARP suppression show and clear commands

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.
`clear ip arp suppression-cache`	Clears the IPv4 ARP suppression cache. You can also clear the cache for a specified VLAN or bridge domain.
`clear ip arp suppression-statistics`	Clears the IPv4 ARP suppression statistical information. You can also clear statistics for a specified VLAN or bridge domain.

NEW! Conversational ARP

Conversational ARP reduces the number of cached ARP entries by programming only active flows into the forwarding plane. This feature helps to optimize the use of hardware resources.

In many scenarios, software requirements for ARP entries are beyond the capacity of the hardware. Conversational ARP limits storage-in-hardware to active ARP entries. Aged-out entries are deleted automatically.

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

Aging-out is also influenced by the enablement and disablement cycle:

- Under conversational ARP, which is disabled by default, a fast-aging policy of 60 seconds (not configurable) applies to all entries in the ARP cache.
- Upon disablement, the conversational ARP timer no longer applies. All current and new entries become permanent.

Static ARPs are not subject to conversational behavior.

> **Note**
> Conversational ARP is supported on SLX 9150 and SLX 9250.

Enable Conversational ARP

Conversational ARP can reduce the number of cached ARP entries, optimizing usage of hardware resources.

Procedure

1. Access global configuration mode.

```
device# configure terminal
```

2. Specify an aging-time value other than the default 300 seconds.

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

This example sets the aging-time value to 600 seconds.

3. Enable conversational ARP.

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

Example

The following example summarizes the commands in this procedure.

```
device# configure terminal
device(config)# host-table aging-time conversational 600
device(config)# host-table aging-mode conversational
```

ARP Show and Clear Commands

You can display and delete Address Resolution Protocol (ARP) entries.

Table 9: ARP show and clear commands

Command	Description
show arp	Displays the ARP entries. You can filter the display by interface or VRF. You can also display only dynamic or only static ARP entries.
show arp access-list	Displays one or all ARP ACLs that are available on a device, including permit statements.
clear arp	Deletes ARP entries, triggering ARP requests for the cleared entries. The **clear arp no-refresh** option clears ARP entries without such requests. You can limit the clearing by interface or VRF. You can also clear for a specified next-hop IP address.

IPv4 Addressing

IPv4 Addressing Overview

IPv4 uses a 32-bit addressing system designed for use in packet-switched networks. IPv4 routing is enabled by default on SLX-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.

An IP address has two sections:

- **Network**—Identifies the network on which the device is configured.
- **Host**—Identifies the host device.

IPv4 does not provide a reliable communication function. No acknowledgments are sent and the only error control is the header checksum. There are no flow-control mechanisms or retransmissions. Internet Control Message Protocol (ICMP) may be used to report any errors.

IP interfaces

SLX-OS devices that operate at Layer 3 allow IP addresses to be configured on the following types of interfaces:

- Ethernet ports
- Virtual routing interfaces
- Loopback interfaces

You can configure up to 128 IP addresses on each interface.

> **Note**
> After you configure a port as a "switchport," you cannot configure Layer 3 interface parameters on that port. The parameters must be configured on the appropriate virtual routing interface.

IP unnumbered interfaces

With large numbers of routers and redundant Layer 3 interfaces, many IP addresses are consumed just to configure the network itself. Using /31 masks reduces the consumption of addresses, but two IP addresses are still consumed per interface. Using unnumbered interfaces greatly reduces the number of IP addresses that are consumed in the configuration of the network.

For more information, see IP unnumbered interface on page 296.

ARP Cache

The ARP cache contains entries that map IP addresses to MAC addresses.

Entries to the Address Resolution Protocol (ARP) cache are added in one of the following ways:

- From devices that are directly attached to the Layer 3 device.
- From an interface-based static IP route that goes to a destination two or more router hops away. The MAC address is either of the destination device of the router interface answering an ARP request on behalf of the device, using proxy ARP.

The ARP cache can contain both dynamic (learned) entries and static (user-configured) entries. The software places an entry in the ARP cache:

- **Dynamic**—When the Layer 3 device learns a device MAC address from an ARP request or ARP reply from the device.
- **Static**—When the interface with proper IP address comes up.

For more information about ARP, see ARP Overview on page 23.

31-bit Subnet Masks on Point-to-Point Networks

To conserve IPv4 address space, you can assign a 31-bit subnet mask 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. They 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, you cannot use network addresses for broadcast purposes. 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 Parameters and Protocols

Most IP parameters are dynamic. They take effect as soon as you run the CLI command. You can verify that a dynamic change has taken effect by displaying the running configuration.

- To display the running configuration, run the **show running-config** command.
- To save a configuration change permanently, save the configuration to the startup configuration file.
- To change the memory allocation, reload the software after you save the changes to the startup configuration file.

The following protocols are disabled by default:

- Route exchange protocols (OSPF, IS-IS, BGP4)
- Multicast protocols (IGMP, PIM-SM)
- Router redundancy protocols (VRRP-E, VRRP)

IP Global Parameters

The following table lists the IP global parameters, their default values, and where to find configuration information.

Table 10: IP global parameters

Parameter	Description	Default
IP state	The Internet Protocol, version 4	Enabled **Note:** You cannot disable IP.
Router ID	The value that routers use to identify themselves to other routers when exchanging route information. OSPF and BGP4 use router IDs to identify routers.	The IP address configured on the lowest-numbered loopback interface. If no loopback interface is configured, then the lowest-numbered IP address configured on the device.
IP Maximum Transmission Unit (MTU)	The maximum length an Ethernet packet can be without being fragmented.	1500 bytes for Ethernet II encapsulation
Address Resolution Protocol (ARP)	A standard IP mechanism that routers use to learn the Media Access Control (MAC) address of a device on the network. The router sends the IP address of a device in the ARP request and receives the device's MAC address in an ARP reply.	Enabled
ARP rate limiting	The maximum number of ARP packets that the device will accept each second. If the device receives more ARP packets than you specify, the device drops the extra ARP packets for the remainder of the one-second interval.	Disabled
ARP age	The maximum time that the device keeps a MAC address in its ARP cache. The device resets the timer to zero when the ARP entry is refreshed and removes the entry if the timer reaches the ARP age. **Note:** You can change the ARP age for an individual interface. See IP Interface Parameters on page 45.	25 minutes
Proxy ARP	An IP mechanism a router can use to answer an ARP request on behalf of a host, by replying with the router's own MAC address instead of the host's.	Enabled
Static ARP entries	An ARP entry that you place in the static ARP table. Static entries do not age out.	No entries
Time to Live (TTL)	The maximum number of routers (hops) through which a packet can pass before being discarded. Each router decreases a packet's TTL by 1 before forwarding the packet. If	64 hops

Table 10: IP global parameters (continued)

Parameter	Description	Default
	decreasing the TTL causes the TTL to be 0, the router drops the packet instead of forwarding it.	
Directed broadcast forwarding	A directed broadcast is a packet containing all ones (or in some cases, all zeros) in the host portion of the destination IP address. When a router forwards such a broadcast, it sends a copy of the packet out each of its enabled IP interfaces. **Note:** You can configure this parameter for an individual interface. See IP Interface Parameters on page 45.	Disabled
Source-routed packet forwarding	A source-routed packet contains a list of IP addresses through which the packet must pass to reach its destination.	Enabled
Internet Control Message Protocol (ICMP) messages	The Extreme device can send the following types of ICMP messages: • Echo messages (ping messages) • Destination Unreachable messages • Redirect messages **Note:** You can configure ICMP Redirect messages for an individual interface. See IP Interface Parameters on page 45.	Enabled
Maximum DHCP relay hops	The maximum number of hops away a BootP server can be located from a router and still be used by the router's clients for network booting.	Four
Domain name for Domain Name Server (DNS) resolver	A domain name (example: extreme.router.com) you can use in place of an IP address for certain operations such as IP pings, trace routes, and Telnet management connections to the device.	None configured
DNS default gateway addresses	A list of gateways attached to the device through which clients attached to the device can reach DNS.	None configured
IP load sharing	A feature that enables the device to balance traffic to a specific destination across multiple equal-cost paths. Load sharing is based on a combination of destination MAC address, source MAC address, destination IP address, source IP address, and IP protocol. **Note:** Load sharing is sometimes called Equal Cost Multi Path (ECMP).	Enabled
Maximum IP load sharing paths	The maximum number of equal-cost paths across which this device is allowed to distribute traffic.	64
Origination of default routes	You can enable a device to originate default routes for the following route exchange protocols, on an individual protocol basis: • OSPF • BGP4 • IS-IS	Disabled
Static route	An IP route that you place in the IP route table.	No entries
Source interface	The IP address the device uses as the source address for Telnet, RADIUS, or TACACS/TACACS+ packets originated by	The lowest-numbered IP

Table 10: IP global parameters (continued)

Parameter	Description	Default
	the device. The device can select the source address based on either of the following: • The lowest-numbered IP address on the interface the packet is sent on. • The lowest-numbered IP address on a specific interface. The address is used as the source for all packets of the specified type regardless of interface the packet is sent on.	address on the interface the packet is sent on.

IP Interface Parameters

The following table lists the interface-level IP parameters, their default values, and where to find configuration information.

Table 11: IP interface parameters

Parameter	Description	Default
IP state	The Internet Protocol, version 4	Enabled **Note:** You cannot disable IP.
IP address	A Layer 3 network interface address The SLX-OS device has separate IP addresses on individual interfaces.	None configured
Encapsulation type	The format of the packets in which the device encapsulates IP datagrams.	Ethernet
IP Maximum Transmission Unit (MTU)	The maximum length (number of bytes) of an encapsulated IP datagram that the device can forward.	1500 for Ethernet II encapsulated packets
ARP age	Locally overrides the global setting. See IP Global Parameters on page 42.	25 minutes
Directed broadcast forwarding	Locally overrides the global setting. See IP Global Parameters on page 42.	Disabled
ICMP Redirect messages	Locally overrides the global setting. See IP Global Parameters on page 42.	Enabled
UDP broadcast forwarding	The device can forward UDP broadcast packets for UDP applications such as BootP. By forwarding the UDP broadcasts, the device enables clients on one subnet to find servers attached to other subnets. **Note:** To enable a client's UDP application request to find a server on another subnet,configure an IP helper address.	The device helps forward broadcasts for the following UDP application protocols: • bootps • dns • netbios-dgm • netbios-ns • tacacs • tftp • time
IP helper address	The IP address of a UDP application server (such as a BootP or DHCP server) or a directed broadcast address. IP helper addresses allow the device to forward requests for certain UDP applications from a client on one subnet to a server on another subnet.	None configured

Assign an IP Address to a Loopback Interface

IP addresses can be assigned to a loopback interface, using Classless Interdomain Routing (CIDR) network masks. Loopback interfaces add stability to a network, because they do not incur route flap problems due to unstable links between devices.

About This Task

IPv4 routing is enabled by default on SLX-OS devices that operate at Layer 3 and cannot be disabled. IP addresses must be assigned to interfaces on the devices to allow IPv4-based protocols to operate across the network.

Procedure

1. From privileged EXEC mode, access global configuration mode.

   ```
   device# configure terminal
   ```

2. Access the interface to which you are assigning the IP addresses.

   ```
   device(config)# interface loopback 1
   ```

3. Assign an IP address to the interface.

   > **Note**
   > You can define only one IP address per loopback. The only valid mask value is **/32**.

   ```
   device(config-Loopback-1)# ip address 1.1.1.1/32
   ```

4. Activate the interface.

   ```
   device(config-Loopback-1)# no shutdown
   ```

5. Verify that the IP address is assigned to the interface.

   ```
   device(config-Loopback-1)# do show ip interface loopback 1
    Loopback 1 is up protocol is up
    Primary Internet Address is 1.1.1.1/32
    IP MTU is 1500
    Proxy Arp is not Enabled
    ICMP unreachables are always sent
    ICMP mask replies are never sent
    IP fast switching is enabled
    Vrf : default-vrf
   ```

Assign IP Addresses to an Ethernet Interface

IP addresses can be assigned to an Ethernet interface, using Classless Interdomain Routing (CIDR) network masks.

Procedure

1. From privileged EXEC mode, access global configuration mode.

   ```
   device# configure terminal
   ```

2. Access the interface to which you are assigning the IP addresses.

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

3. Assign one or more IP addresses, with a CIDR network mask.

   ```
   device(config-if-eth-0/1)# ip address 11.1.1.11/24
   device(config-if-eth-0/1)# ip address 11.11.1.11/24
   ```

4. To assign a secondary IP address, include the **secondary** keyword.

```
device(config-if-eth-0/1)# ip address 10.53.5.4/24 secondary
```

> **Note**
> You can configure a secondary IP address only if the primary IP address is already configured in the same subnet.

5. Activate the interface.

```
device(config-if-eth-0/1)# no shutdown
```

6. Verify that the IP addresses are assigned to the interface.

```
device(config-if-eth-0/1)# do show ip interface ethernet 3/14
 Ethernet 3/14 is up protocol is up
 Primary Internet Address is 11.1.1.11/24 broadcast is 11.1.1.255
 Primary Internet Address is 11.11.1.11/24 broadcast is 11.11.1.255
 Secondary Internet Address is 11.11.1.12/24 broadcast is 11.11.1.255
 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 : default-vrf
```

Delete an IP Address from an Interface

You can delete a specified IP address, or all IP addresses, from an interface.

Procedure

1. From privileged EXEC mode, access global configuration mode.

```
device# configure terminal
```

2. Access the interface from which you are deleting the IP address.

```
device(config)# interface ethernet 1/5
```

3. To delete a specific IP address from the interface, run the **no ip address** command with the IP address and the mask.

```
device(config-if-eth-1/5)# no ip address 10.53.5.3/24
```

4. To delete all IP addresses from the interface, run the **no ip address** command.

```
device(config-if-eth-1/5)# no ip address
```

NEW! Enable IP Directed Broadcast on an Interface

A directed broadcast is an IP broadcast to all devices in a directly attached network or subnet.

About This Task

When a device receives a packet with "Broadcast IP" as the destination address and IP directed broadcast is enabled on a Layer 3 interface, the interface floods the packet to all devices in the directly attached destination network.

IP directed broadcast is supported on default and user-defined VRF.

By default, IP directed broadcast is disabled. You can enable it on a Layer 3 physical Ethernet interface or VE interface. This task uses an Ethernet interface as an example.

Procedure

1. From privileged EXEC mode, access global configuration mode.

   ```
   device# configure terminal
   ```

2. Access interface configuration mode.

   ```
   device(config)# interface ethernet 0/2
   ```

3. Enable IP directed broadcast on the interface.

   ```
   device(config-if-eth-0/2)# ip directed-broadcast
   ```

Router-ID IP Addresses

Certain routing protocols, such as OSPF and BGP4, have algorithms that specify which configured IP address identifies the device.

In general, a device has IP addresses that are assigned to various interfaces. However, some routing protocols identify the device by the router ID, rather than by the IP addresses assigned to the interfaces connected by the protocol. If a router ID is not specified, such protocols use the following algorithm to select a router ID.

1. If the device has loopback interfaces, the router ID is the IP address on the loopback interface with the lowest IP address. For example, if you configure loopback interfaces 1, 2, and 3 as follows, the default router ID is 10.9.9.9/32:

   * Loopback 1: 10.9.9.9/32
   * Loopback 2: 10.4.4.4/32
   * Loopback 3: 10.1.1.1/32

2. If a loopback interface is not configured, then the lowest IP address configured on a physical interface becomes the router ID.

You can also specify a router-id IP address, using the **ip router-id** command.

Assign a Router-ID IP Address

For routing protocols that decide which configured IP address identifies the device, such as OSPF and BGP4, an IP address that you apply under global configuration takes precedence.

About This Task

> **Note**
> If you change the router ID, all current BGP4 sessions are cleared.

Procedure

1. Access global configuration mode.

   ```
   device# configure terminal
   ```

2. Specify the device-level IP address.

```
device(config)# ip router-id 10.11.12.13
```

> **Note**
> You can specify an IP address that is also assigned to an interface on that device. But make sure that the router-ID IP address is not assigned on another network device.

Domain Name System

The Domain Name System (DNS) is 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. A domain name is easier to remember than all of the numbers in an IP address. DNS has several components.

*   **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 in that domain, this device automatically appends the appropriate domain to the host address and forwards it to the DNS server.
*   **DNS resolver**: In a Layer 2 or Layer 3 device, the DNS resolver 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 in 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 that are assigned to the device enable clients that are attached to the device to reach DNS.

Configure a DNS Domain and Gateway Addresses

You can configure a Domain Name System (DNS) domain and DNS gateway addresses to resolve host names to IP addresses.

Procedure

1. Access global configuration mode.

```
device# configure terminal
```

2. Configure a domain name.

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

3. Configure the default DNS gateway address for DNS servers.

```
device(config)# ip dns name-server 10.157.22.199
```

4. Return to privileged EXEC mode.

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

5. (Optional) 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 DNS server (default DNS gateway address), and 10.157.22.80 represents the IP address of the mycompany.com host.

Example

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
```

Source IP address for various packet types

When a device originates a packet of one of the following types, the default source address of the packet is the lowest-numbered IP address on the interface that sends the packet:

- Telnet
- TACACS/TACACS+
- TFTP
- RADIUS
- Syslog
- SNTP

You can configure the device to always use the lowest-numbered IP address on a specific Ethernet, loopback, or virtual interface as the source addresses for these packets. When configured, the device uses the same IP address as the source for all packets of the specified type, regardless of the ports that actually sends the packets.

Designating a source IP address provides the following benefits:

- If your server is configured to accept packets only from specific IP addresses, you can configure the device to always send the packets from the same link or source address.
- If you specify a loopback interface as the single source for specified packets, servers can receive the packets regardless of the states of individual links. Thus, if a link to the server becomes unavailable, but can be reached through another link, the client or server still receives the packets, and the packets still have the source IP address of the loopback interface.

IPv4 Maximum Transmission Unit

The IPv4 maximum transmission unit (MTU) is the maximum length of an IPv4 packet that a Layer 2 frame can contain. IP MTU is supported only on Layer 3 interfaces such as VE port or router port.

If an IPv4 packet is larger than the MTU allowed by the frame, the device fragments the IP packet into multiple parts that fit into frames, and sends the parts of the fragmented IP packet in separate frames.

The device that receives the multiple fragments of the IP packet reassembles the fragments into the original packet. The default IPv4 MTU is 1500 bytes for Ethernet II packets.

There are limitations on MTU. The hardware supports only three profiles, including the default value of 1500 and two user-defined values. The previous lower supported value is used for traffic coming through the device when you configure the MTU above these values.

You can increase the MTU to reduce packet fragmentation. However, MTU cannot be higher than the maximum frame size, minus 18. For more information, see IPv4 MTU and Maximum Frame Size on page 51.

The device supports hardware forwarding for unicast jumbo packets that are received on a port that supports the frame's MTU size and are forwarded to another port that also supports the frame's MTU size.

> **Note**
> For multicast data traffic, frames are not fragmented and the IP MTU setting is ignored.

IPv4 MTU and Maximum Frame Size

Because raising MTU demands system resources, increase MTU only on the IP interfaces that need it.

If you have one interface connected to a server that uses jumbo frames and two other interfaces connected to clients that can support jumbo frames, you might increase the MTU only on those three IP interfaces.

Because you need to allow for the Ethernet header and the cyclic redundancy check (CRC), IP MTU must always be less that the maximum frame size, as follows:

- 18 bytes for untagged packets
- 22 bytes for single-tagged packets
- 26 bytes for dual-tagged packets

Changing the IPv4 MTU on an Interface

You can change the size of the IPv4 maximum transmission unit (MTU) on each Layer 3 interface. IP MTU configuration is not supported on a port-channel.

About This Task

By default, IPv4 MTU is 1500 bytes. Valid values range from 1300 through 9194 bytes.

> **Note**
> If the interface is part of a VE, change the IPv4 MTU only at the VE interface and not at the physical port. Switchports have only the MTU value configured. They do not inherit from the IP MTU of the VE. Extreme recommends that you configure both MTU and IP MTU to be the same value.

Procedure

1. Access global configuration mode.

   ```
   device# configure terminal
   ```

2. To change the IPv4 MTU on an Ethernet interface, take the following steps.

   a. Access the interface on which you want to modify the IPv4 MTU.

   ```
   device(config)# interface ethernet 1/5
   ```

   b. Specify the IPv4 MTU value.

   ```
   device(config-if-eth-1/5)# ip mtu 2000
   ```

3. To change the IPv4 MTU at a VE interface, take the following steps.

   a. Access the VE on which you want to modify the IPv4 MTU.

   ```
   device(config)# interface ve 103
   ```

   b. Specify the IPv4 MTU value.

   ```
   device(config-ve-103)# ip mtu 2000
   ```

IP Addressing Show and Clear Commands

You can display and delete information related to IPv4 interfaces and routes.

Table 12: IP addressing show and clear commands

Command	Description
show ip interface	Displays the IP address, status, and configuration for a specified Ethernet, loopback, or VE interface. You can also display a brief summary of such information for all interfaces.
show ip route	Displays IP route information.
clear ip route	Clears a specified route or all IP routes in the IP routing tables.
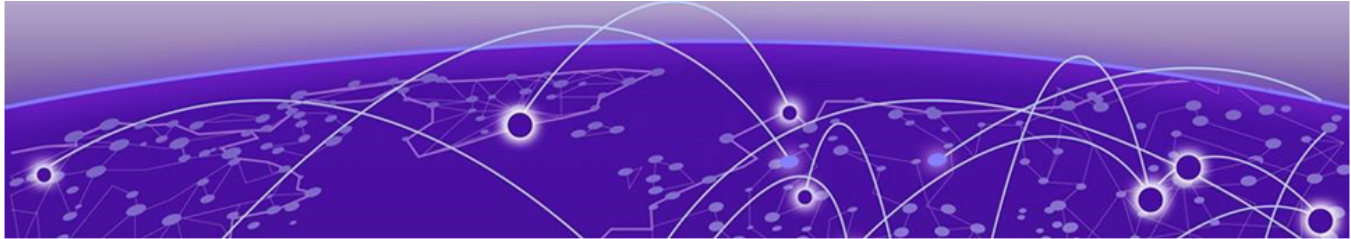
IPv6 Addressing

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 more network devices.

An IPv6 address consists of 8 fields of 16-bit hexadecimal values separated by colons (:).

Network Prefix   Interface ID

HHHH HHHH HHHH HHHH HHHH HHHH HHHH HHHH

128 Bits

HHHH = Hex Value 0000 – FFFF
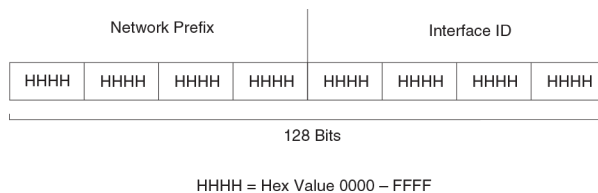
Figure 1: IPv6 address format

As shown in the figure, HHHH is a 16-bit hexadecimal value. H is a 4-bit hexadecimal value. The following is an example of an IPv6 address: `2001:0000:0000:0200:002D:D0FF:FE48:4672`.

An IPv6 address can include 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 the figure, the IPv6 network prefix consists 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, for which the following rules apply.

- 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`.

Configure Global and Link-local IPv6 Addresses

About This Task
Configure global and link-local IPv6 addresses at the interface level. Link-local addresses are not forwarded outside the local network.

Procedure

1. Access global configuration mode.

   ```
   device# configure terminal
   ```

2. Access interface configuration mode.

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

3. Configure a global IPv6 address for the interface.

   ```
   device(conf-if-eth-0/1)# ipv6 address 2001:DB8:12D:1300:240:D0FF:FE48:4672/64
   ```

4. Configure a link-local IPv6 address for the interface.

   ```
   device(config-if-eth-0/1)# ipv6 address FE80::240:D0FF:FE48:4672 link-local
   ```

5. Return to Privileged EXEC mode.

   ```
   device(config-if-eth-0/1)# end
   ```

NEW! Assigning IPv6 addresses to interfaces

You can assign primary and secondary IPv6 addresses to interfaces, using Classless Interdomain Routing (CIDR) network masks.

About This Task

Configuration of IPv6 addresses on an interface has the following conditions.

- Multiple primary IPv6 addresses from different subnets are allowed, but not from the same subnet.
- Secondary IPv6 addresses have the same subnet as the primary addresses.
- Primary IPv6 addresses cannot be deleted when a secondary address is configured.
- Secondary IPv6 addresses cannot be added when the primary address is not configured.

Take the following steps to assign IPv6 addresses. The examples in steps 2-5 are for an Ethernet interface.

Procedure

1. From privileged EXEC mode, enter global configuration mode.

   ```
   device# configure terminal
   ```

2. Access the interface to which you are assigning the IP addresses.

   ```
   device(config)# interface ethernet 0/2
   ```

3. Assign one or more primary IP addresses, including the CIDR network mask.

   ```
   device(config-if-eth-0/2)# ipv6 address 102:102::1/64
   device(config-if-eth-0/2)# ipv6 address 202:102::1/64
   ```

4. Assign one or more secondary IP addresses, including the CIDR network mask and the **secondary** keyword.

   ```
   device(config-if-eth-0/2)# ipv6 address 102:102::2/64 secondary
   device(config-if-eth-0/2)# ipv6 address 202:102::2/64 secondary
   ```

5. Activate the interface.

   ```
   device(config-if-eth-0/2)# no shutdown
   ```

IPv6 management

On the management interface of the SLX-OS device, the IPv6 routing functionality is not enabled.

IPv6 Management ACLs

An Access Control List (ACL) specifies the users, devices, or processes that are allowed access to a particular object.

You can use ACLs to restrict Web management access to only the management functions on an SLX-OS device that is acting as an IPv6 host. Or you can restrict access so that the SLX-OS host can be reached only by a specified IPv6 device.

When you run the **ipv6 access-list** command, the SLX-OS device enters the IPv6 Access List configuration level, where you can access commands for configuring IPv6 ACLs. You can apply ACLs to network management access features such as Telnet, SSH, Web, and SNMP.

> **Note**
> Unlike IPv4, there is no distinction between standard and extended ACLs in IPv6.

The *ACL-name* variable for the command specifies a name for the IPv6 ACL. An IPv6 ACL name cannot start with a numeral, for example, 1access. Also, an IPv4 ACL and an IPv6 ACL cannot share the same name.

For more information about the command, including examples, see the *Extreme SLX-OS Command Reference*.

Specify an IPv6 SNMP Trap Receiver

You can specify an IPv6 host as a trap receiver to ensure that all SNMP traps sent by the device go to the same SNMP trap receiver or set of receivers.

Procedure

1. Access global configuration mode.

   ```
   device# configure terminal
   ```

2. Specify an IPv6 host as a trap receiver.

   ```
   device(config)# snmp-server host ipv6 2001:DB8:89::13
   ```

Secure Shell and IPv6

The Secure Shell (SSH) protocol allows secure remote access to management functions on the device. SSH provides a function similar to Telnet. You can log in to and configure a device using a publicly or commercially available SSH client program, just as you can with Telnet. However, unlike Telnet, which provides no security, SSH provides a secure, encrypted connection to the device.

For related information, see IPv6 Telnet on page 56.

IPv6 Telnet

Telnet sessions can be established between an Extreme device to a remote IPv6 host, and from a remote IPv6 host to the device using IPv6 addresses.

Use the **telnet** command to establish a Telnet connection from an Extreme device to a remote IPv6 host using the console. For example:

```
device# telnet 2001:DB8:3de2:c37::6
```

Up to five read-access Telnet sessions are supported on the router at one time. Write-access through Telnet is limited to one session. Only one outgoing Telnet session is supported on the router at a time.

Use the **show telnet server status** command to view the status of the Telnet sever.

For more information about the commands, including examples, see the *Extreme SLX-OS Command Reference*.

IPv6 Traceroute

The **traceroute** command displays the path of network packets from the SLX-OS device to an IPv6 host.

A traceroute displays information for each hop as soon as the information is received, including the maximum and minimum Time-To-Live values. The device displays up to three responses when there are multiple equal-cost routes to the destination.

For more information about the command, including examples, see the *Extreme SLX-OS Command Reference*.

IPv6 Neighbor Discovery

The Neighbor Discovery feature for IPv6 uses IPv6 ICMP messages to perform the following tasks:

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

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

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

You can adjust the following IPv6 Neighbor Discovery features:

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

> **Note**
> For all solicitation and advertisement messages, SLX-OS uses seconds as the unit of measure instead of milliseconds.

- Amount of time during which an IPv6 node considers a remote node reachable (for use by all nodes on a given link).
- The interval after which the IPv6 Neighbor Discovery cache is deleted or refreshed.

> **Note**
> Neighbor Discovery is not supported on tunnel interfaces.

Router advertisement and solicitation messages

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

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

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

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

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

Configure IPv6 Router Advertisement Preference

IPv6 router advertisement preference improves the ability of IPv6 hosts to select an appropriate router for a remote destination.

About This Task

An IPv6 host uses the preference value in an advertisement to determine the default router to use to reach a destination. A preference value can be low, medium, or high.

Procedure

1. Access global configuration mode.

   ```
   device# configuration terminal
   ```

2. Access interface configuration mode.

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

3. Configure router advertisement preference for the IPv6 router.

   ```
   device(conf-if-eth-0/1)# ipv6 nd router-preference medium
   ```

Configure Router Advertisement of an IPv6 DNS Server

You can configure a maximum of 4 DNS server addresses in an IPv6 router advertisement.

About This Task
The lifetime value is the number of seconds that a DNS server is advertised in a router advertisement. Changes to DNS attributes take effect in the next scheduled router advertisement.

Procedure

1. Access global configuration mode.

   ```
   device# configuration terminal
   ```

2. Access interface configuration mode.

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

3. Configure a DNS server to be advertised.

   ```
   device(conf-if-eth-0/1)# ipv6 nd ra-dns-server 2001:DB8:0:ee44::1 lifetime 200
   ```

   This example uses the **ipv6 nd ra-dns-server** command to configure a DNS server with the 2001:DB8:0:ee44::1 address to be advertised in a router advertisement for 200 seconds.

Configure IPv6 Router Advertisement

You can configure the interval for sending router advertisements, the router advertisement lifetime, and the hop limit.

About This Task

As a best practice, ensure that the interval between router advertisement transmission is less than or equal to the router lifetime value if the router is advertised as a default route.

Procedure

1. Access global configuration mode.

   ```
   device# configure terminal
   ```

2. Access interface configuration mode.

   ```
   device# interface ethernet 0/1
   ```

3. Configure the number of seconds that a router is advertised as the default router on an interface.

   ```
   device(config-if-eth-0/1)# ipv6 nd ra-lifetime 1900
   ```

   If you set the lifetime to 0, the router is not advertised as a default router.

4. Configure the maximum and minimum intervals (in seconds) during which router advertisement messages are sent randomly.

   ```
   device(config-if-eth-0/1)# ipv6 nd ra-interval 1200 min 400
   ```

5. Specify the maximum number of hops to advertise in advertisement messages.

   ```
   device(config-if-eth-0/1)# ipv6 nd hoplimit 32
   ```

6. Specify the size of the maximum transmission unit (MTU) that is advertised in an advertisement message.

   ```
   device(config-if-eth-0/1)# ipv6 nd mtu 2400
   ```

Set Managed Address Flags in IPv6 Router Advertisements

The flags for managed address configuration and other stateful configuration are not set by default in router advertisement messages.

About This Task

Use this procedure to set the following flags.

Flag	Description
Managed Address Configuration flag (**ipv6 nd managed-config-flag**)	Indicates to hosts on a local link that they must use the stateful autoconfiguration feature to obtain IPv6 addresses for their interfaces. Overrides the Other Configuration flag. If this flag is not set, then hosts obtain IPv6 address information as determined by the **ipv6 nd other-config-flag** command.
Other Configuration flag (**ipv6 nd other-config-flag**)	Indicates to hosts on a local link that they can use the stateful autoconfiguration feature to obtain configuration settings other than IPv6 address information for their interfaces.

Procedure

1. Access global configuration mode.

   ```
   device# configuration terminal
   ```

2. Access interface configuration mode.

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

3. Set the Managed Address Configuration flag.

   ```
   device(conf-if-eth-0/1)# ipv6 nd managed-config-flag
   ```

4. Set the Other Configuration flag.

   ```
   device(conf-if-eth-0/1)# ipv6 nd other-config-flag
   ```

Neighbor redirect messages

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

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

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

Duplicate Address Detection (DAD)

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

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

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.

Configure IPv6 Static Neighbor Entries

A static entry in the ND cache causes a neighbor to be reachable at all times

About This Task

If a neighbor cannot be reached by means of Neighbor Discovery (ND), you can add a static entry to the ND cache. A static IPv6 ND entry is like a static IPv4 ARP entry.

```
device(config-if-eth-3/5)# ipv6 neighbor 2001:db8:2678::2 0000.002b.8641
```

Procedure

1. Access global configuration mode.

   ```
   device# configuration terminal
   ```
2. Access interface configuration mode.

   ```
   device (config)# interface ethernet 3/5
   ```
3. Add a static entry for a neighbor.

   This example adds a static entry for a neighbor with IPv6 address `2001:db8:2678::2` and link-layer address `0000.002b.8641`, reachable through interface ethernet 3/5.

   ```
   device(config-if-eth-3/5)# ipv6 neighbor 2001:db8:2678::2 0000.002b.8641
   ```

Configure Reachable Time for Remote IPv6 Nodes

You can configure the duration (in seconds) that a router considers a remote IPv6 node to be reachable.

About This Task

Router advertisement messages include a *reachable time* value. All nodes on a link use the same value. By default, the reachable time duration is 0.

Extreme recommends that you configure a long duration, because a short duration causes IPv6 network devices to process the information at a greater frequency.

Procedure

1. Access global configuration mode.

   ```
   device# configuration terminal
   ```
2. Access interface configuration mode.

   ```
   device(config)# interface ethernet 0/1
   ```
3. Configure the reachable time.

   ```
   device(conf-if-eth-0/1)# ipv6 nd reachable-time 600
   ```

   > **Note**
   > The actual reachable time ranges from 0.5 to 1.5 times the configured or default value.

IPv6 Neighbor Discovery Suppression

In a data center fabric, you can suppress Neighbor Discovery (ND) to help reduce ND control traffic.

> **Note**
> ND suppression is supported on VLANs and bridge domains.

By default, ND is not suppressed, which 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 ND cache.
2. The device broadcasts an ND request throughout the IP fabric.

When ND is suppressed, 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 ND cache.
2. The device broadcasts an 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 only to the device.
   - If there is no match, the leaf control plane broadcasts the request to the entire IP fabric.

> **Tip**
> When the static anycast gateway feature is enabled in an IP fabric, suppress ND on the respective VLANs or BDs to prevent ND broadcast across the network.

Suppress Neighbor Discovery on a VLAN

Use this procedure to suppress Neighbor Discovery (ND) on a VLAN.

Procedure

1. Access global configuration mode.
   ```
   device# configure terminal
   ```
2. Access VLAN configuration mode.
   ```
   device(config)# vlan 110
   ```
3. Suppress ND.
   ```
   device(config-vlan-110)# suppress-nd
   ```

Suppress Neighbor Discovery on a Bridge Domain

Use this procedure to suppress ND suppression on a bridge domain.

Procedure

1. Access global configuration mode.
   ```
   device# configure terminal
   ```
2. Access bridge-domain configuration mode.
   ```
   device(config)# bridge-domain 2
   ```
3. Suppress ND.
   ```
   device(config-bridge-domain-2)# suppress-nd
   ```

Neighbor Discovery Suppression Show and Clear Commands

You can display and delete Neighbor Discovery information.

For more information, including examples, see the *Extreme SLX-OS Command Reference*.

Table 13: ND suppression show and clear commands

Command	Description
`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.
`clear ipv6 nd suppression-cache`	Clears the IPv6 ND suppression cache. You can also clear the cache for a specified VLAN or bridge domain.
`clear ipv6 nd suppression-statistics`	Clears the IPv6 ND suppression statistical information. You can also clear statistics for a specified VLAN or bridge domain.

Conversational Neighbor Discovery

Conversational neighbor discovery (ND) reduces the number of cached ND entries by programming only active flows into the forwarding plane. This feature helps to optimize the use of hardware resources.

In many scenarios, software requirements for ND entries are beyond the capacity of the hardware. Conversational ANDRP limits storage-in-hardware to active ND entries. Aged-out entries are deleted automatically.

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

Aging-out is also influenced by the enablement and disablement cycle:

* Under conversational ND, which is disabled by default, a fast-aging policy of 60 seconds (not configurable) applies to all entries in the ND cache.
* Upon disablement, the conversational ND timer no longer applies. All current and new entries become permanent.

Static NDs are not subject to conversational behavior.

> **Note**
> Conversational ND is supported on SLX 9150 and SLX 9250.

Enable Conversational ND

Conversational ND can reduce the number of cached ND entries, optimizing usage of hardware resources.

Procedure

1. Access global configuration mode.

   ```
   device# configure terminal
   ```

2. Specify an aging-time value other than the default 300 seconds.

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

   This example sets the aging-time value to 600 seconds.

3. Enable conversational ND.

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

Example

The following example summarizes the commands in this procedure.

```
device# configure terminal
device(config)# host-table aging-time conversational 600
device(config)# host-table aging-mode conversational
```

IPv6 Maximum Transmission Unit

The IPv6 maximum transmission unit (MTU) is the maximum length of an IPv6 packet that can be transmitted on an interface.

If an IPv6 packet is longer than an MTU, the host that originated the packet fragments the packet and transmits its contents in multiple packets that are shorter than the MTU.

The hardware supports three MTU profiles: the default value of 1500 and two user-defined values. If the limit is reached, the following error is displayed:
```
%Error: Maximum limit of allowed different IP MTUs reached.
```

The hardware supports one MTU value at the interface.

Change the IPv6 MTU Value

About This Task

You can change the MTU value globally and at the interface level. The value at the interface has precedence over the global value.

Procedure

1. Access global configuration mode.

   ```
   device# configure terminal
   ```

2. Change the MTU globally.

   ```
   device(config)# ip mtu 2000
   ```

   The `ip mtu` command configures both IPv4 and IPv6 MTU.

3. Access interface Ethernet configuration mode.

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

4. Change the MTU for the interface.

```
device(conf-if-eth-0/1)# ip mtu 1300
```

IPv6 Prefix List

An IPv6 prefix list consists of one or more conditional statements that pose an action (permit or deny) if a route matches a specified prefix.

In prefix lists with multiple statements, you can specify a sequence number for each statement. The specified sequence number determines the order in which the statement appears in the prefix.

You can configure an IPv6 prefix list on a global basis and use it as input to other commands or processes, such as route aggregation, route redistribution, route distribution, and route maps. When a device sends or receives an IPv6 route, it applies the statements in the IPv6 prefix list in their order of appearance to the packet. When a match occurs, the device takes the specified action (permit or deny the packet) and stops further comparison for that route.

You can use permit statements in the prefix list to specify the route that you want to send to the other feature. If you use deny statements, the route specified by the deny statements is not supplied to the other feature.

A device supports IPv6 prefix lists, which you can use for basic route filtering. You can configure up to 100 IPv6 prefix lists.

You must specify the ipv6-prefix parameter in hexadecimal 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-prefix parameter and precede the prefix-length parameter.

Configure an IPv6 Prefix List

You can configure an IPv6 prefix list for basic traffic filtering.

Procedure

1. Access global configuration mode.

```
device# configure terminal
```

2. Configure an IPv6 prefix list.

```
(config)# ipv6 prefix-list prefix-filter-1 permit FE80::/10 ge 25 le 25
```

The **ge** *ge-value* or **le** *le-value* that you specify must meet the following condition for prefix-length: `ge-value <= le-value <= 128`.

If you do not specify **ge** *ge-value* or **le** *le-value*, the prefix list matches only on the exact prefix you specify with the ipv6-prefix/prefix-length parameter.

Display Prefix List Information

You can use a show command to display the contents of an IPv6 prefix list.

About This Task

You can run the command from any level of the CLI.

Procedure

Display the contents of a specified prefix list.

```
device# show ipv6 prefix-list routesfor2001
ipv6 prefix-list routesfor2001: 2 entries
    seq 5 permit 2001::/16
    seq 10 permit 2001:db8::/32
```

Display Global IPv6 Information

You can use show commands to display information about IPv6 interfaces, neighbors, and route tables.

Procedure

1. Display IPv6 interface information.

```
device# show ipv6 interface brief

Interface                        Vrf                    Status         Protocol
        IPv6-Address
================================================================================
Loopback 1                       default-vrf            up             up
        3911::4/128
Ethernet 0/1                     default-vrf            up             up
        3002::4/64

Ve 2                             default-vrf            up             down
        a1a1:0:2::2/64
Ve 3                             default-vrf            up             down
        a1a1:0:3::2/64
Ve 4                             default-vrf            up             down
        a1a1:0:4::2/64
```

2. Display IPv6 neighbor information.

```
device# show ipv6 neighbor
Address                                        Mac-address     Interface
MacResolved    Age         Type
-----------------------------------------------------------------------------------
--------------------
a1a1:0:3::1                                    609c.9f02.1f15  Ve 3
no             10:25:32    Dynamic
fe80::21b:edff:fe9f:1900                       001b.ed9f.1900  Eth 0/1
yes            00:02:36    Dynamic
fe80::629c:9fff:fe02:1f15                      609c.9f02.1f15  Ve 2
no             10:25:32    Dynamic
fe80::629c:9fff:fe02:1f15                      609c.9f02.1f15  Ve 3
no             10:25:42    Dynamic
fe80::629c:9fff:fe02:1f15                      609c.9f02.1f15  Ve 4
no             10:32:04    Dynamic
fe80::629c:9fff:fe02:1f15                      609c.9f02.1f15  Ve 5
no             10:32:05    Dynamic
```

3. Display the IPv6 route table.

```
device# show ipv6 route
IPv6 Routing Table for VRF "default-vrf"
Total number of IPv6 routes: 5
'*' denotes best ucast next-hop
'[x/y]' denotes [preference/metric]

3002::/64, attached
    *via ::, Eth 1/1, [0/0], 10h19m, direct, tag 0
3002::4/128, attached
    *via ::, Eth 1/1, [0/0], 10h19m, local, tag 0
3911::4/128, attached
    *via ::, Lo 1, [0/0], 12h6m, direct, tag 0
fe80::/10, attached
    *via ::, , [0/0], 12h6m, local, tag 0
ff00::/8, attached
    *via ::, Null0, [0/0], 12h6m, local, tag 0
```

Clear Global IPv6 Information

You can use clear commands to remove IPv6 neighbor discovery information and IPv6 routes.

Procedure

1. Clear the IPv6 neighbor discovery cache on an interface.

```
device# clear ipv6 neighbor 2000:7838::1 force-delete
```

2. Clear the IPv6 routing tables on an interface.

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

Prevention of DDoS attack on IPv6 Subnet-router Anycast Addresses

To prevent a Distributed Denial of Service (DDoS) attack, you can drop traffic that is destined for IPv6 subnet-router anycast addresses.

RFC 4291 requires that all devices support an IPv6 subnet-router anycast address, which is defined by an all-zero subnet prefix. Traffic sent to an IPv6 subnet-router anycast address is delivered to one device on the subnet. Devices are required to support subnet-router anycast addresses for the subnets to which they have interfaces.

The IPv6 subnet-router anycast address is used by applications in which a node needs to communicate with any one of the set of devices. However, this functionality creates a vulnerability whereby a DDoS attack can send traffic to the IPv6 subnet-router anycast address.

You can use the **ipv6 subnet-zero drop** command to drop traffic that is destined for the IPv6 subnet-router anycast address. For more information, see Drop IPv6 Subnet-router Anycast Address Traffic on page 68.

Drop IPv6 Subnet-router Anycast Address Traffic

You can enable hardware to drop IPv6 subnet-router anycast address traffic that has the all-zero subnet prefix.

About This Task

By default, traffic destined for IPv6 subnet-router anycast addresses is allowed. However, you can deny the traffic for reasons such as preventing a DDoS attack. For more information, see Prevention of DDoS attack on IPv6 Subnet-router Anycast Addresses on page 67.

Procedure

1. Access global configuration mode.

   ```
   device# configure terminal
   ```

2. Enable hardware to drop traffic destined for IPv6 subnet-router anycast addresses.

   ```
   device(config)# ipv6 subnet-zero drop
   ```

IPv4 Static Routing

Overview of IPv4 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 and can use routing information learned from other protocols.

The IPv4 route table can receive routes from several sources, such as static routes, directly connected networks, OSPF, BGP4, and ISIS protocols.

Static routes are manually configured entries in the IPv4 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
- Next-hop router
- Ethernet interface, typically used for directly attached destination networks
- Port-channel interface
- Virtual interface
- Null interface

You can influence the preference a route is given:

- Configure a route metric higher than the default metric

- Give the route an administrative distance
- Specify 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

IPv4 Static Route Availability

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.

If the interface is not available and the next-hop IP address is invalid, the software removes the static route from the route table. When the port or virtual routing interface becomes available and the next-hop is valid, the software adds the route 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 2: Example of static route

In the example, the static route to the 207.95.7.0 destination was configured as follows, 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 in that subnet are on the same port. Switch A deduces that IP interface 207.95.7.7 is also on port 1/2.

Configure a Basic IPv4 Static Route

Specify the IPv4 destination address, the address mask, and the IPv4 address of the next hop.

Procedure

1. Access global configuration mode.

```
device# configure terminal
```

2. Enter the IP address and prefix length for the route destination network and the IP address for the next hop.

```
device(config)# ip route 10.0.0.0/24 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 in the Class C subnet address in the destination IP address.

NEW! Enable Recursive Lookup for an IPv4 Static Route

The recursive lookup feature allows a device to search for another route if the original next-hop is not reachable.

About This Task

If the next-hop for a basic static route is directly reachable, it is added to the routing table. If that next-hop is not reachable, the device can perform a lookup for another route (a recursive route). You can enable the recursive lookup feature in the default VRF or in a non-default VRF.

Procedure

1. Access global configuration mode.

```
device# configure terminal
```

2. To enable recursive lookup in the default VRF, run the following command.

```
device(config)# ip route next-hop-recursion
```

3. To enable recursive lookup in a non-default VRF, run the following commands.

```
device(config) vrf red
device(config-vrf-red)# address-family ipv4 unicast
device(config-vrf-red-ipv4-unicast)ip route next-hop-recursion
```

This example enables recursive lookup in a VRF named red.

Add a Cost Metric or Administrative Distance to an IPv4 Static Route

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

About This Task

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

Procedure

1. Access global configuration mode.

```
device# configure terminal
```

2. Designate the route destination, the next hop, and the route priority.

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)# 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.

Example

This example configures a static route with an administrative distance of 3.

```
device(config)# ip route 10.0.0.0/24 10.111.10.1 distance 3
```

Example

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

```
device(config)# ip route 10.128.2.69/24 10.111.10.1 2
```

NEW! Configure a Next Hop for an IPv4 Static Route

You can configure a physical interface, a virtual interface, or a VRF as a next hop.

About This Task

Table 14: Configuration considerations

Interface type	Description
Physical interface	• The interface for the 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. • ARP is generated for a forwarded packet destination IP address when a physical interface is configured as the next hop.
Virtual interface	The interface for the 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.
VRF	• The VRF must be an existing VRF. Any physical port referenced in the next hop must have a valid IP address so that the route is added to the routing table. • 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.

Procedure

1. Access global configuration mode.

   ```
   device# configure terminal
   ```

2. To configure a physical interface, enter the destination IP address and prefix length, the **ethernet** keyword, and the interface number of the next hop.

   ```
   device(config)# ip route 10.128.2.69/24 ethernet 1/4
   ```

3. To configure a virtual interface, enter the destination IP address and prefix length, the **ve** keyword, and the VLAN ID.

   ```
   device(config)# ip route 10.128.2.0/24 ve 3
   ```

4. To configure a VRF, enter the destination IP address and prefix length, the **next-hop-vrf** keyword, the VRF name, and then identify the next hop as an IP address, interface, or null route.

   ```
   device(config)# ip route 10.0.0.0/24 next-hop-vrf red ethernet 1/1
   ```

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

Example

This example configures an IP static route to destination network addresses beginning with 10.0.0.0 through the next-hop interface 2/1.

```
device(config)# ip route 10.0.0.0/24 ethernet 2/1
```

Example

This 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(config)# ip route 10.128.2.0/24 ve 3
```

Example

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

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

Example

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

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

Example

This example discards the traffic for IP address 10.0.0.0/24 through VRF red using a null route.

```
device(config)# ip route 10.0.0.0/24 next-hop-vrf red null 0
```

Configure an IPv4 Static Route to Use with a Route Map

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

Procedure

1. Access global configuration mode.

    ```
    device# configure terminal
    ```

2. Enter the destination network IP address, prefix-length, next-hop IP address, the **tag** keyword, and a tag number.

    ```
    device(config)# ip route 10.0.0.0/24 10.1.1.1 tag 3
    ```

    The tag "3" in this example can be used in a route map.

Configure a Null Static Route

A null static route sends network traffic to a null interface, where the network packets are discarded.

About This Task

You can use a null static route to handle traffic that should not be forwarded because the preferred route is unavailable. The following figure depicts how a null static route works with a standard route to the same destination.
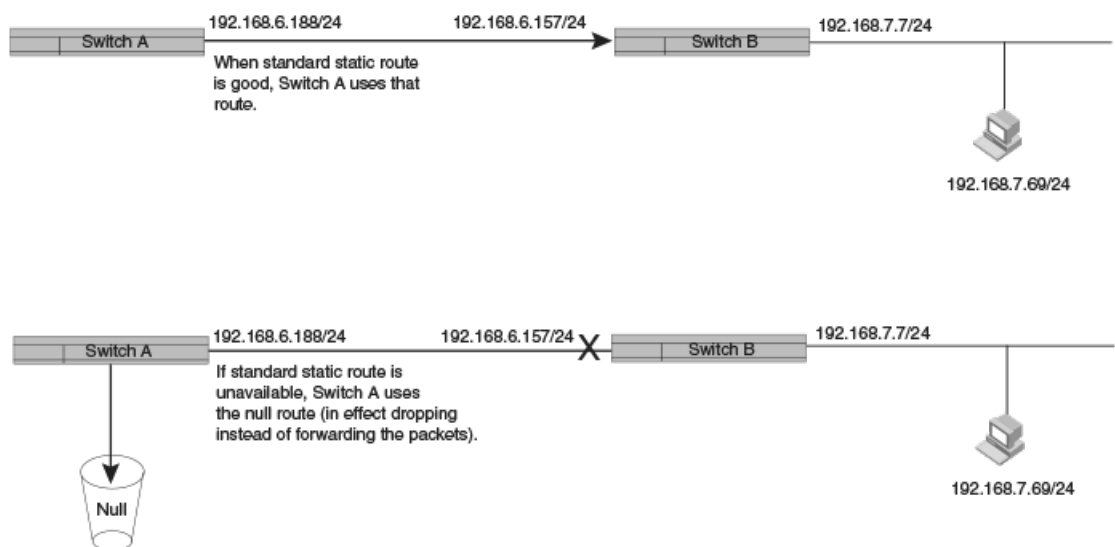
Figure 3: Null route and standard route to same destination

Procedure

1. Access global configuration mode.

   ```
   device# configure terminal
   ```

2. Configure the preferred route to a destination.

   ```
   device(config)# 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.

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

   ```
   device(config)# 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 higher so that the preferred route is used if it is available. When the preferred route becomes unavailable, the null route is used, and traffic to the destination is dropped.

Example

The following example summarizes the commands in this procedure.

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

Configure an IPv4 Static Route

A router uses a default static route when there are no other default routes to a destination.

About This Task

You cannot create a default route to a virtual or physical interface.

Procedure

1. Access global configuration mode.

   ```
   device# configure terminal
   ```

2. Enter the destination route and prefix-length (0.0.0.0/0) followed by a valid next hop IP address.

   ```
   device(config)# ip route 0.0.0.0/0 10.24.4.1
   ```

Example

This example configures a default route that is a null route.

```
device# configure terminal
device(config)# ip route 0.0.0.0/0 null 0
```

Configure an IPv4 Static Route Between Two Named VRFs

You can configure the next hop of a static route to be in a different VRF.

Before You Begin

Ensure that VRFs in the route configuration are valid VRFs.

Procedure

1. Access global configuration mode.

```
device# configure terminal
```

2. Enter configuration mode for the source VRF.

```
device(config)# vrf red
```

In this example, the source VRF is "red."

3. Create a Route Descriptor.

```
device(config-vrf-red)# rd 1:1
```

4. Specify that IPv4 addressing is to be used on the route.

```
device(config-vrf-red)# address-family ipv4 unicast
```

5. Configure the destination address and mask, the next hop VRF, and its IP address.

```
device(config-vrf-red-ipv4-unicast)# ip route 10.128.2.69/24 next-hop-vrf blue 10.1.1.1
```

Configure an IPv4 Static Interface Route Across VRFs

You can configure an IPv4 static interface route from one VRF to an IPv4 interface in a different VRF.

Before You Begin
Ensure that VRFs in the route configuration are valid VRFs.

About This Task
By configuring static routes between VRFs, you can connect one VRF to a host that is directly connected to a port in a different VRF. You can do this by configuring a static route to the interface that is directly connected to the device with the IP address you want to reach.

Consider VRF A and VRF B with the following characteristics. The steps in this procedure create an interface route between these two sample VRFs.

	VRF A	VRF B
Route Distinguisher	1:1	2:2
Interface	Ethernet port 1/1	Ethernet port 1/2
IP Address	10.0.0.1/24	10.0.0.1/24

Procedure

1. Configure VRF A.

   a. Define a route descriptor for VRF A in VRF configuration mode.

   ```
   device# configure terminal
   device(config)# vrf A
   ```

```
device(config-vrf-A)# rd 1:1
```

b. Configure VRF A to use IPv4 addressing.

```
device(config-vrf-A)# address-family ipv4 unicast
device(config-vrf-A-ipv4-unicast)# exit
device(config-vrf-A)# exit
```

c. Configure a device interface to forward traffic over VRF A.

```
device(config)# interface ethernet 1/1
device(config-if-e10000-1/1)# vrf forwarding A
```

d. Assign an IP address to the interface.

```
device(config-if-e10000-1/1)# ip address 10.0.0.1/24
device(config-if-e10000-1/1)# exit
device(config)#
```

2. Configure VRF B.

a. Define a route descriptor for VRF B in VRF configuration mode.

```
device(config)# vrf b
device(config-vrf-b)# rd 2:2
```

b. Configure VRF B to use IPv4 addressing.

```
device(config-vrf-b)# address-family ipv4 unicast
device(config-vrf-b-ipv4-unicast)# exit
device(config-vrf-b)# exit
```

c. Configure an interface to forward traffic from VRF B.

```
device(config)# interface ethernet 1/2
device(config-if-e10000-1/2)# vrf forwarding B
```

d. Assign an IP address to the interface.

```
device(config-if-e10000-1/2)# ip address 10.0.0.1/24
device(config-if-e10000-1/2)# exit
```

3. Configure an IP static route for VRF A that directs traffic to the port that forwards traffic from VRF B.

```
device(config)# vrf a
device(config-vrf-A)# ip route 10.0.0.1/24 ethernet 1/2
```

Example

The following is the complete process for configuring VRF A to forward traffic over port 1/1 and VRF B to forward traffic over port 1/2. Port 1/2 is configured with an IP address of 10.0.0.1/24.The last step configures a static route on VRF A to use port 1/2 as the next-hop to network destinations beginning with 10.0.0.1/24, thereby connecting VRF A to VRF B.

```
device# configure terminal
device(config)# vrf A
device(config-vrf-A)# rd 1:1
device(config-vrf-A)# address-family ipv4 unicast
```

```
device(config-vrf-A-ipv4-unicast)# exit
device(config-vrf-A)# exit
device(config)# interface ethernet 1/1
device(config-if-e10000-1/1)# vrf forwarding A
device(config-if-e10000-1/1)# ip address 10.0.0.1/24
device(config-if-e10000-1/1)# exit
device(config)# vrf B
device(config-vrf-B)# rd 2:2
device(config-vrf-B)# address-family ipv4 unicast
device(config-vrf-B-ipv4u)# exit
device(config-vrf-B)# exit
device(config)# interface ethernet 1/2
device(config-if-e10000-1/2)# vrf forwarding B
device(config-if-e10000-1/2)# ip address 10.0.0.1/24
device(config-if-e10000-1/2)# exit
device(config)# vrf a
device(config-vrf-A)# ip route 10.0.0.1/24 ethernet 1/2
```

Configure IPv4 Static Routes for Load Sharing and Redundancy

You can configure multiple static routes to the same destination as load sharing or backup routes.

About This Task

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.
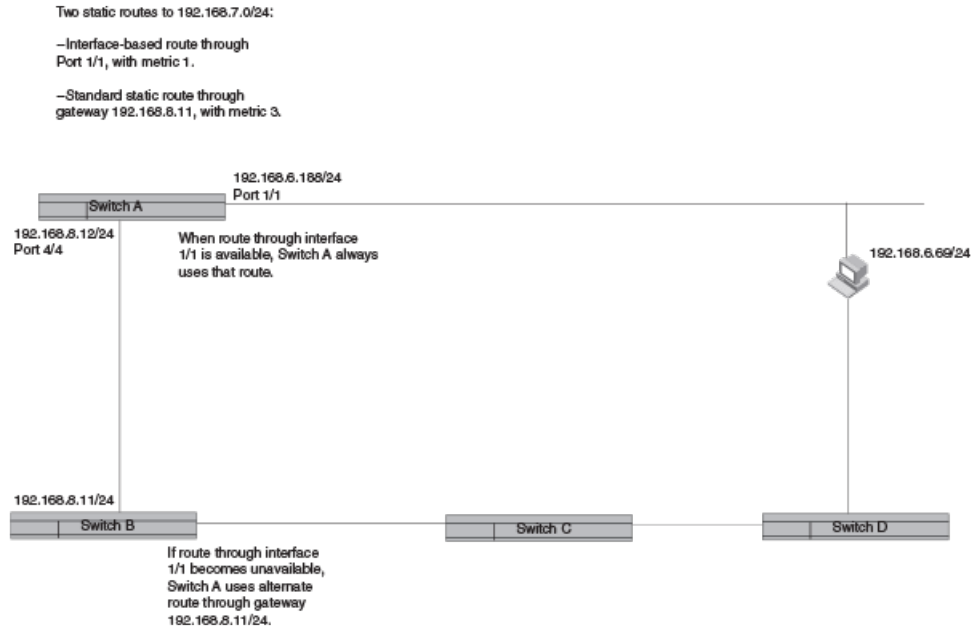
Figure 4: Two static routes to same destination

> **Note**
> You can also use administrative distance to set route priority. Assign the static route a lower administrative distance than other types of routes, unless you want the other route types to be preferred over the static route.

Procedure

1. Access global configuration mode.

```
device# configure terminal
```

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

```
device(config)# ip route 10.128.2.0/24 10.157.22.1
device(config)# ip route 10.128.2.0/24 10.111.10.1
device(config)# ip route 10.128.2.0/24 10.1.1.1
```

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

3. To prioritize the routes, use different metrics for each possible next hop.

```
device(config)# ip route 10.128.2.0/24 10.157.22.1
device(config)# ip route 10.128.2.0/24 10.111.10.1 2
device(config)# 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.

Display IPv4 Static Route Information

You can use show commands to display information about 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.

Procedure

1. Display the configured static routes.

```
device# show running-config ip route
ip route 0.0.0.0/0 11.1.1.20
ip route 0.0.0.0/0 12.1.1.20
ip route 11.16.1.0/24 11.1.1.20
ip route 11.12.1.0/24 null 0 2 distance 4 tag 10
ip route 11.13.1.0/24 null 0
ip route 11.14.1.0/24 ethernet 3/14
ip route 11.15.1.0/24 ve 10
ip route 11.17.1.0/24 ethernet 3/14 4 distance 5 tag 6
device#
```

2. Display a list of active static routes and their connection times.

```
device# show ip route static
IP Routing Table for VRF "default-vrf"
Total number of IP routes: 8
'*' denotes best ucast next-hop
'[x/y]' denotes [preference/metric]

0.0.0.0/0
    *via 11.1.1.20, Eth 3/14, [1/1], 5m32s, static, tag 0
11.12.1.0/24
    *via DIRECT, Null0, [4/2], 16m57s, static, tag 10
11.13.1.0/24
    *via DIRECT, Null0, [1/1], 16m12s, static, tag 0
11.14.1.0/24
    *via DIRECT, Eth 3/14, [1/1], 11m58s, static, tag 0
11.16.1.0/24
    *via 11.1.1.20, Eth 3/14, [1/1], 9m46s, static, tag 0
11.17.1.0/24
    *via DIRECT, Eth 3/14, [5/4], 9m2s, static, tag 6
device#
```

3. Display all active IP routes and their connection times.

```
device# show ip route
IP Routing Table for VRF "default-vrf"
Total number of IP routes: 8
'*' denotes best ucast next-hop
'[x/y]' denotes [preference/metric]

0.0.0.0/0
    *via 11.1.1.20, Eth 3/14, [1/1], 5m19s, static, tag 0
11.1.1.0/24, attached
    *via DIRECT, Eth 3/14, [0/0], 17m46s, direct, tag 0
11.1.1.11/32, attached
    *via DIRECT, Eth 3/14, [0/0], 17m46s, local, tag 0
11.12.1.0/24
    *via DIRECT, Null0, [4/2], 16m44s, static, tag 10
11.13.1.0/24
    *via DIRECT, Null0, [1/1], 15m59s, static, tag 0
11.14.1.0/24
```

```
    *via DIRECT, Eth 3/14, [1/1], 11m45s, static, tag 0
11.16.1.0/24
    *via 11.1.1.20, Eth 3/14, [1/1], 9m33s, static, tag 0
11.17.1.0/24
    *via DIRECT, Eth 3/14, [5/4], 8m49s, static, tag 6
device#
```

IPv6 Static Routing

Overview of IPv6 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 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
- Next hop router
- VRF for the next hop
- Ethernet interface, typically used for directly attached destination networks
- Port-channel interface
- Virtual interface
- Null interface

You can influence the preference a route is given:

- Configure a route metric higher than the default metric
- Give 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

IPv6 Static Route Availability

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.

If the interface is not available and the next-hop IP address is invalid, the software removes the static route from the route table. When the port or virtual routing interface becomes available and the next-hop is valid, the software adds the route 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 5: Example of static route

In the example, the static route to 2001:DB8::0/32 was configured as follows, 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 in that subnet are on the same port. Switch A deduces that IP interface 2001:DB8::0/32 is also on port 1/2.

Configure a Basic IPv6 Static Route

Specify the IPv6 destination address, the address mask, and the IPv6 address of the next hop.

Before You Begin
Enable IPv6 on at least one interface by configuring an IPv6 address or explicitly enabling IPv6 on that interface.

Procedure

1. Access global configuration mode.

   ```
   device# configure terminal
   ```

2. Enter the route destination IPv6 address in hexadecimal with 16-bit values between colons, the address prefix length preceded by a slash, and the IPv6 address of the next-hop gateway.

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

   > **Note**
   > The IPv6 address architecture is defined in RFC 2373.

NEW! Enable Recursive Lookup for an IPv6 Static Route

The recursive lookup feature allows a device to search for another route if the original next-hop is not reachable.

About This Task

If the next-hop for a basic static route is directly reachable, it is added to the routing table. If that next-hop is not reachable, the device can perform a lookup for another route (a recursive route). You can enable the recursive lookup feature in the default VRF or in a non-default VRF.

Procedure

1. Access global configuration mode.

   ```
   device# configure terminal
   ```

2. To enable recursive lookup in the default VRF, run the following command.

   ```
   device(config)# ipv6 route next-hop-recursion
   ```

3. To enable recursive lookup in a non-default VRF, run the following commands.

   ```
   device(config) vrf red
   device(config-vrf-red)# address-family ipv6 unicast
   device(config-vrf-red-ipv6-unicast)ipv6 route next-hop-recursion
   ```

   This example enables recursive lookup in a VRF named red.

Add a Cost Metric or Administrative Distance to an IPv6 Static Route

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

Before You Begin

Enable IPv6 on at least one interface by configuring an IPv6 address or explicitly enabling IPv6 on that interface.

Procedure

1. Access global configuration mode.

   ```
   device# configure terminal
   ```

2. Designate the route destination, the next hop, and the route priority.

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)# ipv6 route 2001:DB8::0/64 2001:DB8:2343:0:ee44::1 2
```

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

Example

This example configures a static route with an administrative distance of 3.

```
device(config)# ipv6 route 2001:DB8::0/64 2001:DB8:0:ee44::1 distance 3
```

Example

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

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

NEW! # Configure a Next Hop for an IPv6 Static Route

You can configure a physical interface, a virtual interface, or a non-default VRF as a next hop.

About This Task

Table 15: Configuration considerations

Interface type	Description
Physical interface Virtual interface	The interface for the 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.
VRF	• The VRF must be an existing VRF. Any physical port referenced in the next hop must have a valid IP address so that the route is added to the routing table. • The VRF must be a non-default VRF.

Procedure

1. Access global configuration mode.

```
device# configure terminal
```

2. To configure a physical interface:

   a. Enter the destination IP address, the **ethernet** keyword, and the interface number of the next hop.

   ```
   device(config)# ipv6 route 2001:DB8::0/32 fe80::1 ethernet 3/1
   ```

   This 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 Ethernet interface 3/1.

3. To configure a virtual interface:

   a. Enter the destination IP address, the **ve** keyword, and the VLAN ID.

   ```
   device(config)# ipv6 route 2001:DB8::0/32 fe80::1 ve 3
   ```

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

4. To configure a VRF:

   a. Enter the destination IP address, the **next-hop-vrf** keyword, and the name of the VRF that contains the next-hop gateway router and its IPv6 address.

   ```
   device(config)# 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.

Configure an IPv6 Static Route to Use with a Route Map

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

Before You Begin

Enable IPv6 on at least one interface by configuring an IPv6 address or explicitly enabling IPv6 on that interface.

Procedure

1. Access global configuration mode.

   ```
   device# configure terminal
   ```

2. Enter the destination IP address, next-hop address, the **tag** keyword, and a tag number.

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

   The tag "3" in this example can be used in a route map.

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.

About This Task

> **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.
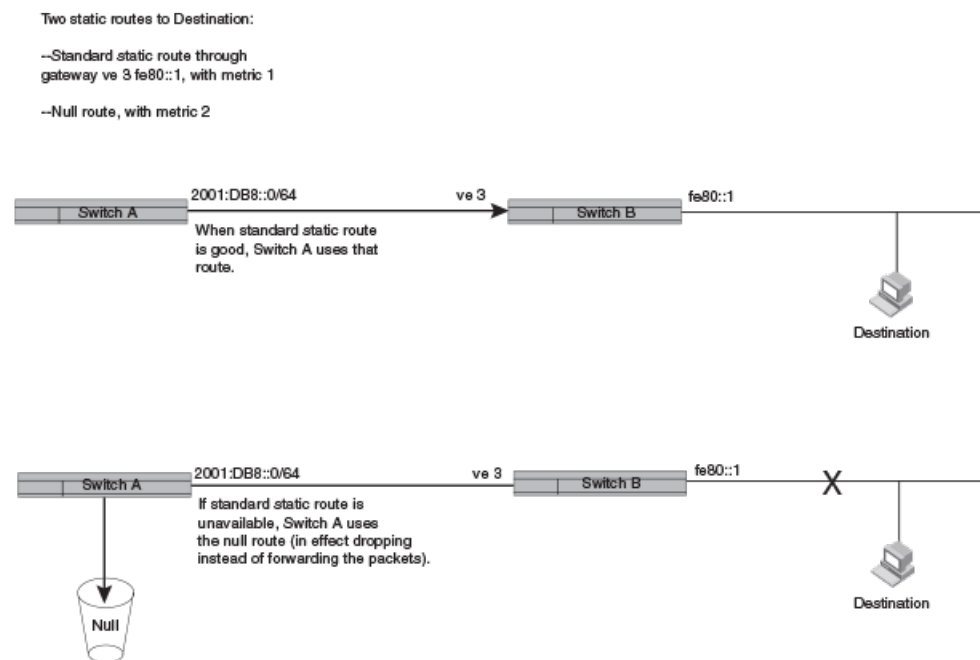
Figure 6: Null route and standard route to same destination

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.

Procedure

1. Enter global configuration mode.

```
device# configure terminal
```

2. Configure the preferred route to a destination.

```
device(config)# ipv6 route 2001:DB8::0/64 fe80::1 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.

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

```
device(config)# 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.

Example

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)# ipv6 route 2001 : DB8 : : 0/64 fe80::1 ve 3
device(config)# ipv6 route 2001 : DB8 : : 0/64 null 0 2
```

Configure an IPv6 Static Route

A router uses a default static route when there are no other default routes to a destination.

Procedure

1. Access global configuration mode.

```
device# configure terminal
```

2. Enter the destination route and network mask (::/0) followed by a valid next-hop IP address.

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

Configure IPv6 Static Routes for Load Sharing and Redundancy

You can configure multiple static routes to the same destination as load sharing or backup routes.

About This Task

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.

Two static routes to 2001:DB8::0/64:

--Primary static route through gateway 2001:1:DB8:2343:0:ee44::1, with default metric 1.

--Standard static route through gateway 2001:DB8:2344:0:ee44::2, with metric 2.
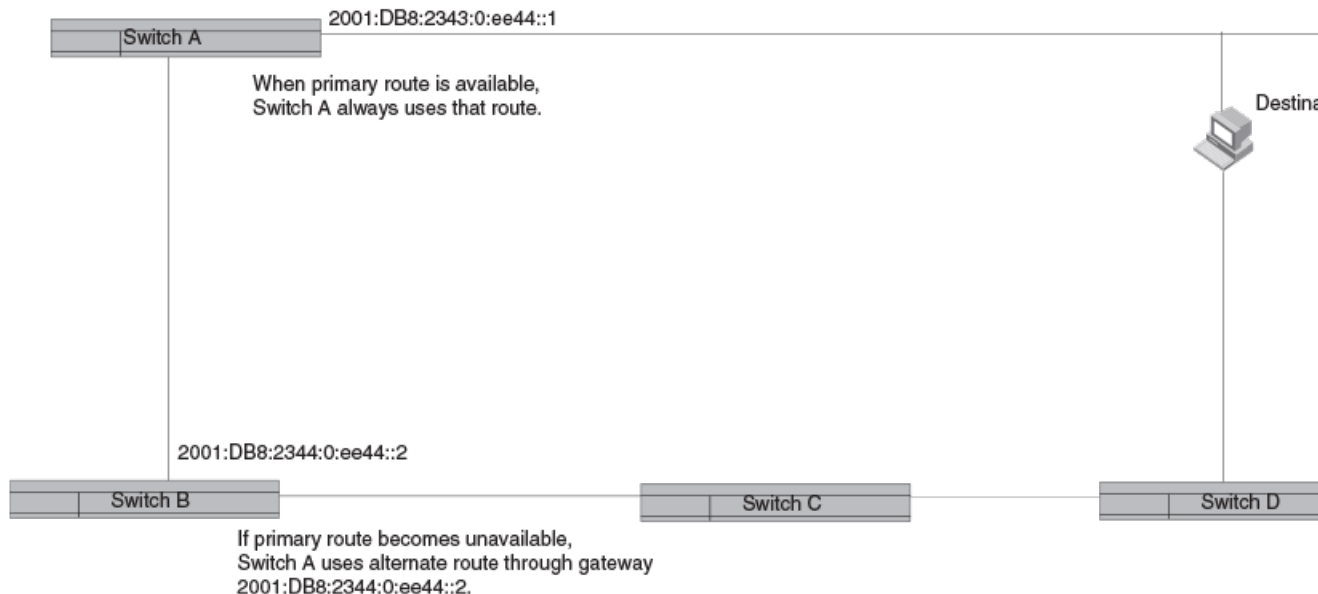
Figure 7: Two static routes to same destination

> **Note**
> You can also use administrative distance to set route priority. Assign the static route a lower administrative distance than other types of routes, unless you want the other route types to be preferred over the static route.

Procedure

1. Access global configuration mode.

   ```
   device# configure terminal
   ```

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

   ```
   device(config)# ipv6 route 2001:DB8::0/64 2001:DB8:2343:0:ee44::1
   device(config)# 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 alternates between the two paths.

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

```
device(config)# ipv6 route 2001:DB8::0/64 2001:DB8:2343:0:ee44::1
device(config)# 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 (the default metric is not entered in the CLI). If this path is not available, traffic is directed through 2001:DB8:2344:0:ee44::2, which has the next lowest metric of 2.

Remove an IPv6 Static Route

Use the no form of the **ipv6 route** command to remove a static route.

Procedure

1. (Optional) View configured routes and confirm parameters,

```
device# show ipv6 route
IPv6 Routing Table for VRF "default-vrf"
Total number of IPv6 routes: 11
'*' denotes best ucast next-hop
'[x/y]' denotes [preference/metric]

1200:1201::/64, attached
    *via ::, Eth 2/45, [0/0], 45m29s, direct, tag 0
1200:1201::1:1/128, attached
    *via ::, Eth 2/45, [0/0], 45m29s, local, tag 0
1200:1202::/64, attached
    *via ::, Ve 2, [0/0], 45m26s, direct, tag 0
1200:1202::1:1/128, attached
    *via ::, Ve 2, [0/0], 45m26s, local, tag 0
2221::/32
    *via 1200:1201::1:2, Eth 2/45, [100/10], 11m41s, static, tag 300
2222::/48
    *via fe80::205:33ff:fee6:a531, Eth 2/45, [1/1], 43m44s, static, tag 0
2222::1/128
    *via fe80::205:33ff:fee6:a531, Eth 2/45, [110/1], 0m7s, ospfv3, intra, tag 0
2223::/64
    *via 1200:1202::1:2, Ve 2, [1/1], 3m45s, static, tag 0
2224::1/128
    *via fe80::205:33ff:fee6:a501, Ve 2, [1/1], 43m41s, static, tag 0
fe80::/10, attached
    *via ::, , [0/0], 6h30m, local, tag 0
ff00::/8, attached
    *via ::, Null0, [0/0], 6h30m, local, tag 0
```

2. (Optional) Narrow the output to static routes only.

```
device# show ipv6 route static
IPv6 Configured Static Routes for VRF "default-vrf"

3002:7::/64-> 1200:3::1:2 preference: 1
   nh_vrf (default-vrf)

3002:9::/64-> 1200:4::1:2 preference: 1
   nh_vrf (default-vrf)
```

3. Access global configuration mode.

```
device# configure terminal
```

4. Remove the static route, including the destination and next hop.

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

You do not need to include cost metric, distance, or tag parameters.

Remove an IPv6 Static Route in a Non-default VRF

Use the no form of the **ipv6 route** command to remove a static route.

Procedure

1. (Optional) View configured routes and confirm parameters.

```
device# show ipv6 route 2225::/32 vrf blue
IPv6 Routing Table for VRF "blue"
Total number of IPv6 routes: 10
'*' denotes best ucast next-hop
'[x/y]' denotes [preference/metric]

2225::/32
    *via 1200:3::1:2, Eth 2/47, [1/1], 0m5s, static, tag 0

device# show ipv6 route 2225::/32 vrf blue
Total number of IP routes: 0
device#
```

2. Access VRF configuration mode.

```
device# configure terminal
device(config)# vrf blue
device(config-vrf-blue)# address-family ipv6 unicast
device(vrf-blue-ipv6-unicast)#
```

3. Remove the static route, including the destination and next hop.

```
device(vrf-blue-ipv6-unicast)# no ipv6 route 2225::/32 1200:3::1:2
```

You do not need to include cost metric, distance, or tag parameters.

4. (Optional) Confirm that the route was removed.

```
device# show ipv6 route 2225::/32 vrf blue
Total number of IP routes: 0
device#
```

Display IPv6 Static Route Information

You can use show commands to display information about connected, static, and protocol routes.

Procedure

1. Display the IPv6 route table information.

```
device# show ipv6 route
IPv6 Routing Table for VRF "default-vrf"
Total number of IPv6 routes: 11
'*' denotes best ucast next-hop
```

```
'[x/y]' denotes [preference/metric]

1200:1201::/64, attached
    *via ::, Eth 2/45, [0/0], 45m29s, direct, tag 0
1200:1201::1:1/128, attached
    *via ::, Eth 2/45, [0/0], 45m29s, local, tag 0
1200:1202::/64, attached
    *via ::, Ve 2, [0/0], 45m26s, direct, tag 0
1200:1202::1:1/128, attached
    *via ::, Ve 2, [0/0], 45m26s, local, tag 0
2221::/32
    *via 1200:1201::1:2, Eth 2/45, [100/10], 11m41s, static, tag 300
2222::/48
    *via fe80::205:33ff:fee6:a531, Eth 2/45, [1/1], 43m44s, static, tag 0
2222::1/128
    *via fe80::205:33ff:fee6:a531, Eth 2/45, [110/1], 0m7s, ospfv3, intra, tag 0
2223::/64
    *via 1200:1202::1:2, Ve 2, [1/1], 3m45s, static, tag 0
2224::1/128
    *via fe80::205:33ff:fee6:a501, Ve 2, [1/1], 43m41s, static, tag 0
fe80::/10, attached
    *via ::, , [0/0], 6h30m, local, tag 0
ff00::/8, attached
    *via ::, Null0, [0/0], 6h30m, local, tag 0
```

2. Narrow the output to only static routes.

```
device# show ipv6 static route
IPv6 Configured Static Routes for VRF "default-vrf"

3002:7::/64-> 1200:3::1:2 preference: 1
   nh_vrf (default-vrf)

3002:9::/64-> 1200:4::1:2 preference: 1
   nh_vrf (default-vrf)
device#
```
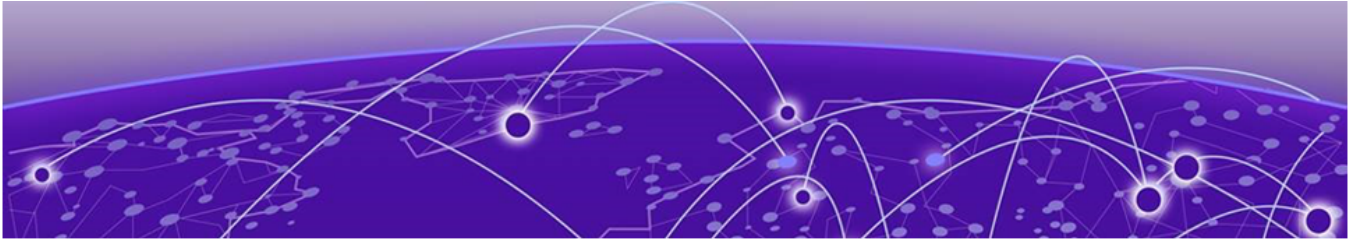
DHCPv4

DHCPv4 Overview

The Dynamic Host Configuration Protocol for IPv4 (DHCPv4) enables DHCP servers to pass configuration parameters to IPv4 hosts.

DHCPv4 is based on the Bootstrap Protocol and provides configuration parameters to DHCP clients on request:

- IP addresses
- Default routes
- DNS server addresses
- Access control
- QoS policies
- Security policies in DHCP server databases

DHCP enables the automatic configuration of client systems. DHCP removes the need to configure devices individually. Instead, clients set network properties by connecting to the DHCP server. 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 has a client-server model, where designated DHCP server hosts allocate network addresses and deliver configuration parameters to dynamically configured hosts. DHCP uses a four-message exchange between client and server: DISCOVER, OFFER, REQUEST, and ACKNOWLEDGE.

DHCP uses MAC addresses to identify a client and uses broadcast IPv4 messages on the LAN.

When you enable DHCP on a router, the router can automatically obtain an Ethernet IP address, prefix length, and default gateway address from the DHCP server.

DHCPv4 Relay Overview

The DHCPv4 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 DHCPv4 relay feature on any Layer 3 interface, such as a switch front-end Ethernet interface (VE port) or a physical interface.

The following supported configuration examples are supported:

* 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 DHCP relay agent forwards DHCP Bootstrap Protocol (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.
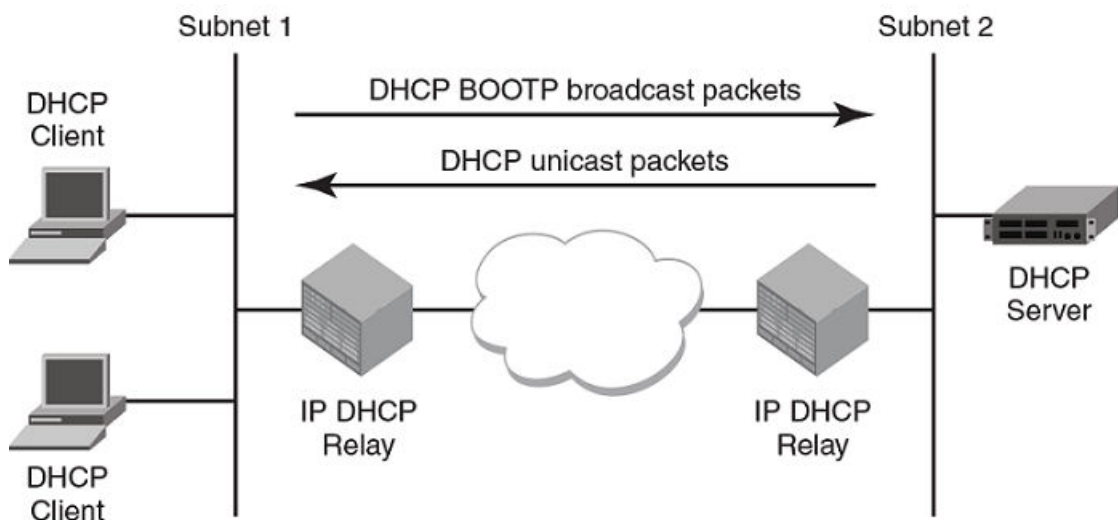
Example

A relay agent is configured on a VE interface, VE 100, which acts over two physical interfaces, Ethernet 1/1 and Ethernet 1/2, corresponding to two client circuits, C1 and C2. Broadcast replies from the server are forwarded to both the circuits, irrespective of the circuit from which the client packet is received by the relay agent, when both the circuits belong to the same broadcast network.
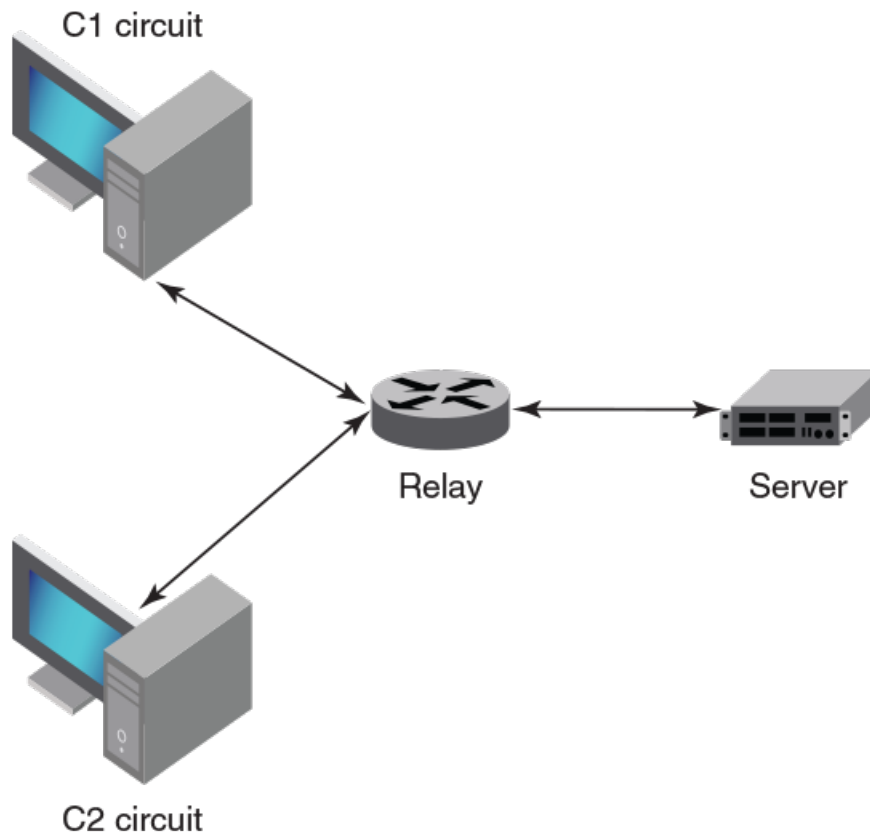
Figure 8: IP DHCP relay function

In this scenario, C1 is connected to Ethernet 1/1 and C2 is connected to Ethernet 1/2 physical interfaces. The relay agent is configured at VE 100. VE 100 is configured on VLAN 100, where Ethernet 1/1 and 1/2 ports are tagged. In such a configuration, broadcast responses from the server are received by both C1 and C2.

Configure the DHCPv4 Relay

You can configure the DHCPv4 relay feature on any Layer 3 interface, such as a switch front-end Ethernet interface (VE port) or a physical interface.

About This Task

The DHCPv4 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.

Consider the following when you configure the IP DHCP relay agent:

- You can configure up to 16 DHCP server IP addresses per interface. When multiple addresses are configured, the relay agent relays the packets to all server addresses. The total number of addresses configurable on the router is 4000.
- The DHCP server and the clients it communicates with can be attached to different VRF instances. For more information, see VRF Support in DHCPv4 on page 100.

Procedure

1. Access global configuration mode.

```
device# configure
```

2. Access interface configuration mode.

```
device(config)# interface ethernet 1/18
```

3. Identify the DHCP server to which DHCP client requests are to be forwarded.

   a. If the DHCP server and its clients are attached to the same VRF instance, use the following syntax.

   ```
   device(config-if-eth-1/18)# ip dhcp relay address 100.1.1.2
   ```

   b. If the DHCP server and its clients are on different VRF instances, use the following syntax.

   ```
   (config)# int ve 100
   device(config-ve-100)# ip dhcp relay 100.1.1.2 use-vrf blue
   ```

Example

This example removes a relay address.

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

Example

This example removes a relay address when the DHCP server is on a different VRF instance.

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

DHCPv4 Relay Supported Scenarios

The following examples illustrate the network environments where the DHCPv4 relay agent is supported.
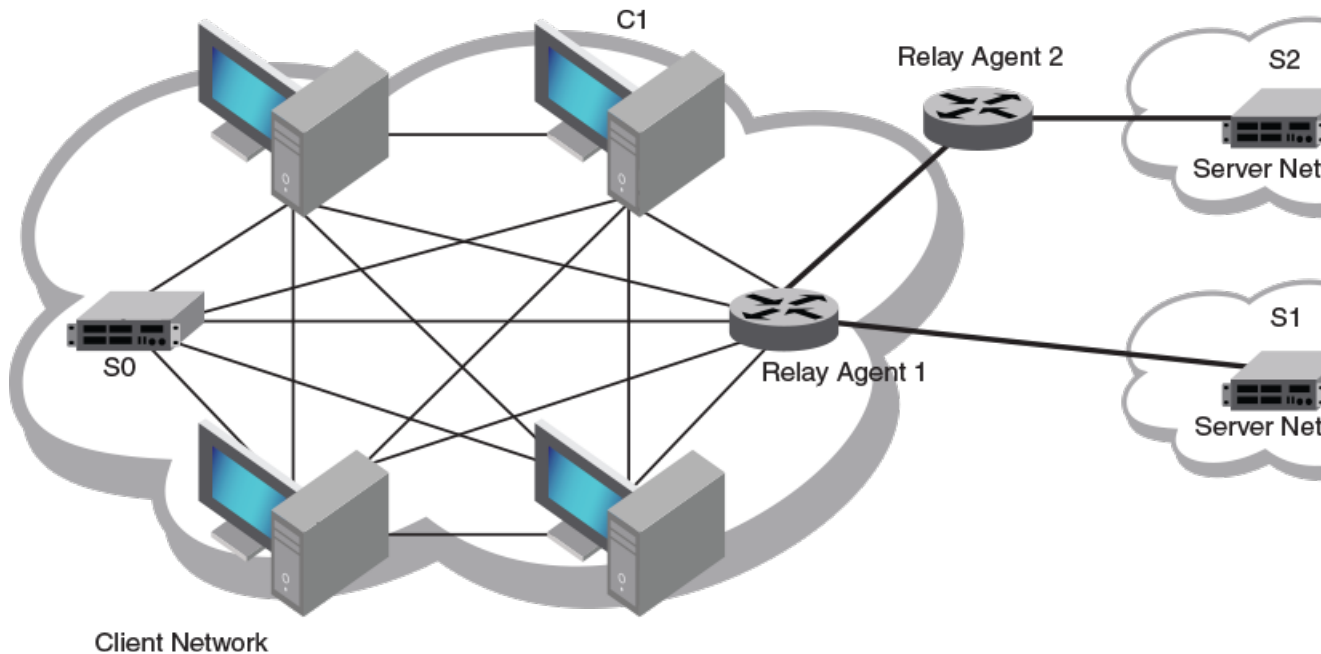
Figure 9: DHCP relay agent user scenarios

DHCP server local

> Client 1 and Server 0 (S0) are on the same subnet (same broadcast domain). A relay agent is not required in this scenario, because the server receives broadcast discover messages from the client directly. A configured relay agent in this scenario is redundant because the relay agent relays (unicast) the received DHCP packet again to the destination server or client. As a result, both parties receive multiple packets.

DHCP server remote

> Client 1 (C1) and Server 1 (S1) are on different subnets but directly linked through a relay agent.

DHCP servers in a network

> Client 1 (C1) and Server 2 (S2) are on different subnets and connected through relay chaining (relay multi-hops).

Multiple DHCP servers

> Server 1 (S1) and Server 2 (S2) are two server addresses on different subnets configured at relay agent 1.

DHCP server in different VRFs

> Client 1 (C1) and Server 1 (S1) are on different VRFs.

DHCPv4 Relay Agent Option 82

DHCP option 82 is a security feature that enables the relay agent to prevent DHCP client requests from untrusted sources. You can configure the relay agent to add option 82 information to DHCP requests from clients before forwarding the requests to the DHCP server.

Option 82 overview

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.

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

Consider the following when you configure option 82:

- If the relay agent is configured over a Ve interface, the remote-id will be the ifindex of the Ve interface, and the broadcast replies from the server are flooded to all the tagged interfaces configured in the Ve.
- 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.

- 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.

Option 82 sub-options

The DHCP Relay Agent Information Option is a container option for specific agent-supplied sub-options. The relay agent information option has the following format.

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.

Table 16: Agent remote ID sub-option

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

Table 17: 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 " Extremenetworks" is used in the circuit ID.

Enable DHCPv4 Relay Agent Option 82

DHCP option 82 is a security feature that enables the relay agent to prevent DHCP client requests from untrusted sources.

About This Task

For more information, see DHCPv4 Relay Agent Option 82 on page 98.

Procedure

1. Access global configuration mode.

```
device# configure
```

2. Enable option 82.

```
device(config)# ip dhcp relay information option
```

Configure the DHCPv4 Relay Gateway Address

When the DHCP relay agent forwards a Bootstrap Protocol or DHCP request, the relay agent sets the Gateway Address field (GIADDR) of the packet with the address you configure.

About This Task

Consider the following when you configure the DHCPv4 relay gateway address.

- If the relay gateway address is configured, but no IP address is configured on the interface, the DHCP relay agent does not relay the requests.
- When the relay gateway address matches an IP address that is configured on the interface, it is used to stamp the request.
- When the relay gateway address does not match the IP addresses configured on the interface, the DHCP relay agent uses the default value, which is the lowest-numbered IP address configured on the interface.
- Virtual IP addresses or local subnet broadcast addresses must not be used as the relay gateway address.
- Any primary or secondary IP address configured on the interface can be used as the relay gateway address.

Procedure

1. Access global configuration mode.

```
device# configure terminal
```

2. Access interface configuration mode.

```
device(config)# interface ethernet 1/4
```

3. Enter the IP address to be used in the Gateway Address field of Bootstrap Protocol or DHCP requests received on a Layer 3 interface.

```
device(config-if-eth-1/4)# ip dhcp relay gateway 10.50.22.26
```

The relay agent places the IP address 10.50.22.26 in the Gateway Address field of Bootstrap Protocol or DHCP requests that the router receives on port 1/4 and forwards to the BootP or DHCP server.

VRF Support in DHCPv4

Virtual Routing and Forwarding (VRF) controls information flow in a network by partitioning the network into different logical VRF domains to isolate traffic.

One device can have multiple containers of routing tables or Forwarding Information Bases (FIBs), with one routing table for each VRF instance. A VRF-capable router can 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 one DHCP server (which may be on a different VRF) so that multiple VRFs can communicate with that server.

DHCP relay is supported in the following configurations.

Same VRF instance as the interface through which the client is connected

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)

Different VRF instance as the interface through which the client is connected, an inter-VRF deployment

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 inter-VRF IP DHCP Relay address configurations is allowed per node. A VRF route leak configuration is required for these configurations. In this 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 <exit interface/next-hop-ip>
- ip route 12.3.4.6/32 next-hop-vrf red <exit interface/next-hop-ip>

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.

> **Note**
> As a best practice, do not configure the same DHCP relay address on different VRFs. For 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"

Display DHCPv4 Relay Information

You can use show commands to display information about DHCP relay statistics, relay addresses for a device, and relay addresses for an interface.

Procedure

1. Access privileged EXEC mode on a device where DHCP relay is configured.
2. Display statistics such as the number of client packets and the number of dropped server packets for the device.

```
device# show ip dhcp relay statistics

Address    Disc.   Offer  Req.   Ack   Nak   Decline    Inform
-------    -----   -----  ----   ---   ---   -------    ------
2.3.4.5     300     100   1211   1201   0       0          0
10.0.1.2    300     100   1211   1207   0       0          0

Client Packets: 2701
```

```
Server Packets: 2932
Client Packets Dropped: 0
Server Packets Dropped: 0
```

3. Display all DHCP relay addresses configured on the device.

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

---------                  -------------              --------
Interface                  Relay Address             VRF Name
---------                  -------------              --------
eth1/18                    105.0.0.10                default-vrf
eth1/18                    107.0.0.1                 default-vrf
ve 200                     107.0.0.1                 default-vrf
```

4. Display all DHCP relay addresses configured for a specific interface on the device.

```
device# show ip dhcp relay address interface ethernet 1/18
DHCP Relay Information Option: Enabled
DHCP Remote ID (Type:Length:Vlan:Mac): 00:08:0000:000000f8003e
---------                  -------------              --------
Interface                  Relay Address             VRF Name
---------                  -------------              --------
eth 1/18                    105.0.0.10                 default-vrf
eth 1/18                    107.0.0.1                  default-vrf
```

This example displays addresses configured on interface 1/18.

```
device# show ip dhcp relay address interface ve 200
DHCP Relay Information Option: Enabled
DHCP Remote ID (Type:Length:Vlan:Mac): 00:08:0000:000000f8003d
---------                  -------------              --------
Interface                  Relay Address             VRF Name
---------                  -------------              --------
ve 200                      107.0.0.1                  default-vrf
```

This example displays addresses configured on VE 200.

Clear DHCPv4 Relay Statistics

You can use clear commands to remove DHCP relay statistics for a specific IP address on the device or all IP addresses on the device.

Procedure

1. Access privileged EXEC mode on a device where DHCP relay is configured.
2. Clear the relay statistics for a specific IP address on the device.

```
device# clear ip dhcp relay statistics ip-address 105.0.0.10
```

3. Clear the relay statistics for all IP addresses on the device.

```
device# clear ip dhcp relay statistics
```

DHCPv6

DHCPv6 Overview

The Dynamic Host Configuration Protocol for IPv6 (DHCPv6) enables DHCP servers to pass configuration parameters to IPv6 nodes.

DHCPv6 offers the capability of automatic allocation of reusable network addresses and additional configuration flexibility.

DHCPv6 has a client-server model, where designated DHCPv6 server hosts allocate network addresses and deliver configuration parameters to dynamically configured hosts. DHCPv6 uses a four-message exchange between client and server: SOLICIT, ADVERTISE, REQUEST, and REPLY.

DHCPv6 identifies a client by its unique identifier and uses Router Advertisement (RA) and IPv6 multicast messages.

The following scalability numbers apply to DHCPv6 relay:

- The total number of DHCPv6 server addresses is 4000.
- The total number of DHCPv6 server addresses per interface is 16.
- The total number of inter-vrf Client-Server configurations for DHCPv4 and v6 relay is 128.

DHCPv6 Relay Agent

A Dynamic Host Configuration Protocol for IPv6 (DHCPv6) relay agent relays messages between the client and the server.

A client locates a DHCPv6 server using a reserved, link-scoped multicast address. Direct communication between the client and server requires them to be 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 that is 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 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 in 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 several 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 on the same VRF as the interface on which the client resides or on a different VRF. The list of destination addresses includes unicast IPv6 addresses, the All_DHCP_Servers multicast address, or other user-defined IPv6 multicast group address.

DHCPv6 prefix delegation

The DHCPv6 relay agent relays all the DHCPv6 messages (which can contain the DHCPv6 options) that are to be relayed across the DHCPv6 client and the server. The relay agent does not access or extract the information in the prefix delegation option.

Relay chaining

DHCPv6 messages can be relayed through multiple relay agents. The Relay-Reply message from the server is 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 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.

> **Note**
> The **show ipv6 dhcp relay statistics** command does not display the packet count in the statistics for DHCPv6 Advertise and Reconfigure messages, because it is sent from the DHCPv6 server to DHCPv6 client directly, not through the relay server.

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 leases are not validated, only 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 leases are not validated, only 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, or 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 is not using 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. 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 (13)

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.

The table lists the DHCPv4 and DHCPv6 message types:

Table 18: 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

Configure DHCPv6 Relay

Configure the IPv6 DHCP relay agent on any Layer 3 interface using the IPv6 address of the DHCP server where client requests are to be forwarded.

About This Task

Layer 3 interfaces can be virtual Ethernet (VE) or physical interfaces. You can configure up to 16 relay destination addresses per interface.

Use the **use-vrf** *vrf-name* parameters if the DHCP server and client interface are on different Virtual Forwarding and Routing (VRF) instances. For more information, see the **ipv6 dhcp relay address** command in the *Extreme SLX-OS Command Reference*.

Procedure

1. Access global configuration mode.

   ```
   device# configure
   ```

2. Access interface configuration mode for the interface where you want to configure the relay.

   ```
   device# interface ethernet 1/17
   ```

3. Enter the IP address of the DHCP server followed by the interface number.

   ```
   device(config-if-eth-1/17# ipv6 dhcp relay address fe80::1016 interface ethernet 2/67
   ```

Example

This example configures a DHCPv6 relay address on a VE. The DHCP server and the client interface are on different VRF instances.

```
device# config
device(config)# interface ve 100
device(config-ve-100)# ipv6 dhcp relay address 2001::1122:AABB:CCDD:3344 use-vrf blue
```

Display DHCPv6 Relay Information

You can use show commands to display information about DHCP relay statistics, relay addresses for a device, and relay addresses for an interface.

Procedure

1. Access privileged EXEC mode on a device where DHCPv6 relay is configured.

2. Display all DHCP relay addresses configured on the device.

   ```
   device# show ipv6 dhcp relay address
   DHCPv6 unique identifier(DUID): 0102f8f10027f8d43dfb
   Interface          Relay Address          VRF Name            Outgoing Interface
   ---------          -------------          --------            ------------------
   eth 1/17           2002::10               default-vrf
   eth 1/17           2000::10               default-vrf         eth 0/9
   eth 1/17           2019::10               test                ve 102
   ve 94              2011::10               default-vrf
   ```

3. Display all DHCP relay addresses configured for a specific interface on the device.

   ```
   device# show ipv6 dhcp relay address interface ethernet 1/40
   DHCPv6 unique identifier(DUID): 01021025768ef804e005
   Interface          Relay Address           VRF Name              Outgoing Interface
   ```

```
---------            -------------           --------            ------------------
Eth 1/40             2019::10                default-vrf
```

4. Display statistics such as the number of packets dropped, received, and sent.

```
device# show ipv6 dhcp relay statistics
DHCPv6 Relay Statistics
-----------------------
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
```

Clear DHCPv6 Relay Statistics

You can use clear commands to remove DHCP relay statistics for a specific IP address on the device or all IP addresses on the device.

Procedure

1. Access privileged EXEC mode on a device where DHCP relay is configured.
2. Clear the relay statistics for a specific IP address on the device.

```
device# clear ipv6 dhcp relay statistics ip-address 2000::10
```

3. Clear the relay statistics for all IP addresses on the device.

```
device# clear ipv6 dhcp relay statistics
```

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.

> **Note**
> Hardware-assisted BFD helps support multiple BFD sessions.
>
> On SLX 9150 and SLX 9250 series devices, hardware-assisted BFD is supported over IPv4 and IPv6.
>
> On SLX 9540 and SLX 9640 series devices, hardware-assisted BFD is not supported over IPv6, but supports:
> - IPv4 single-hop
> - IPv4 multihop, up to a maximum of 16 multihop sessions
> - Session over physical interface, virtual Ethernet (VE) interface, and Link Aggregation Group (LAG)

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. After 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. After 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-upon frequency from the neighbor. If the agreed-upon messages are not received from the neighbor during 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 and multihop routed paths. 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.

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.
> - 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 Configuration Considerations

There are several general points to consider when configuring Bidirectional Forwarding Detection (BFD).

- BFD 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.
- SLX devices support 250 BFD sessions. For a list of supported devices, see Supported Hardware on page 22

For more information, see Considerations for Configuring BFD for Layer 3 Protocols on page 112

BFD for Layer 3 Protocols

Layer 3 protocols can use Bidirectional Forwarding Detection (BFD) 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

- IS-IS
- BGP4
- BGP4+

BFD must be enabled at the interface and routing protocol levels. BFD asynchronous mode depends on the sending of BFD control packets between two systems to activate and maintain BFD neighbor sessions between routers. Therefore, BFD must be configured on both BFD peers.

A BFD session is created after BFD is enabled on the interfaces and at the router level for the appropriate routing protocols. BFD timers are then negotiated, and the BFD peers begin to send BFD control packets to each other at the negotiated interval.

BFD provides one point of forwarding path monitoring when more than one Layer 3 application wants to monitor a host. BFD runs a 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.

> **Note**
> BFD, IS-IS, and OSPF stop operating when Rapid Spanning Tree Protocol (RSTP) path-cost changes are made to the Alt Discarding port on the switch.

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.
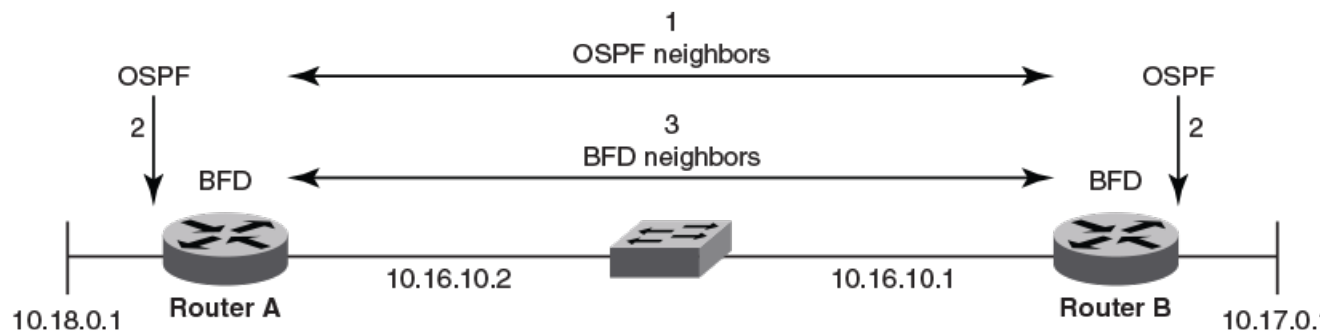
1. OSPF discovers a neighbor.
2. OSPF requests that the local BFD process initiate a BFD neighbor session with the OSPF neighbor router.
3. A BFD neighbor session is established with the OSPF neighbor router.

Figure 10: Establishing a BFD neighbor session

The following figure shows the termination of a BFD neighbor session after a failure occurs in the network.
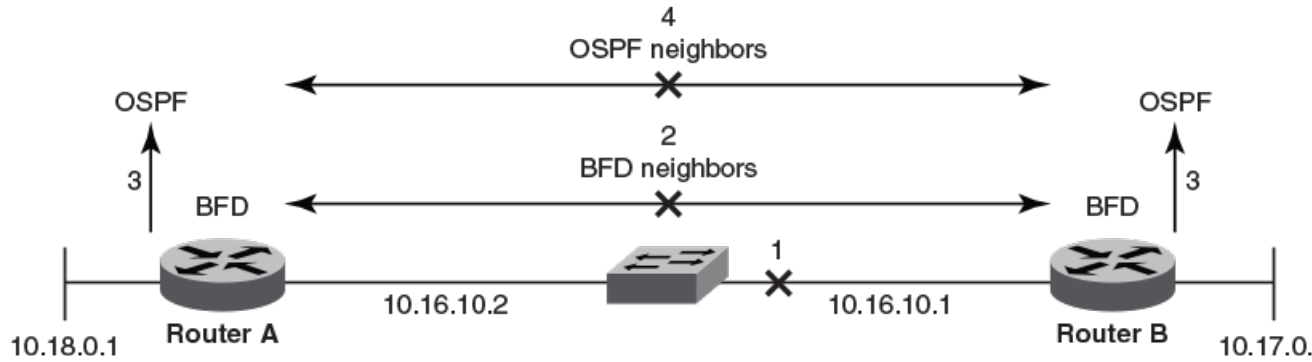
1. A network failure occurs.
2. The BFD session with the OSPF neigbor router is torn down.
3. BFD notifies the local OSPF process that the BFD neighbor is not reachable.
4. The local OSPF process tears down the OSPF relationship and starts reconverging.

Figure 11: Terminating a BFD neighbor session

Considerations for Configuring BFD for Layer 3 Protocols

There are several points to consider when configuring BFD for Layer 3 protocols.

* BFD supports 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 the Layer 3 protocol is notified of any change to the source IP address of the BFD control packet.
* Multicast or anycast address IP addresses cannot be used as source IP addresses.
* BFD establishes only one BFD session per data protocol path (IPv4 or IPv6) regardless of the number of protocols that want to use it.
* If a Layer 3 interface is configured, BFD is enabled by default on the interface (IPv4 or IPv6).
* 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. These sessions are in the Admin Down state and are transmitted at a slower rate than configured values.
* 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 holdover interval expires. If BFD declares a session up before this interval expires, no action is taken by the protocol.
* If you unconfigure a BFD session that is in the up state, OSPF or BGP tells BFD to delete the session and set the reason as Admin Down. Upon receiving 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 a Remote Admin Down event to the routing protocols.

BFD for Layer 3 Protocols on VE Interfaces

Bidirectional Forwarding Detection (BFD) can be configured for Layer 3 protocols on virtual Ethernet (VE) interfaces.

When BFD for Layer 3 protocols is configured on VE interfaces, one of the physical ports is chosen to set up the session. This physical port is allocated after Address Resolution Protocol (ARP) is resolved for that neighbor. For more information, see ARP Overview on page 23.

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 during the BFD detection time, the session is declared as down.

Configure BFD Sessions on an Interface

You can configure the intervals that a device waits to send (transmit) or receive control packets from Bidirectional Forwarding Detection (BFD) peers.

About This Task
BFD is enabled by default on configured Layer 3 interfaces (IPv4 or IPv6). Take the following steps to configure the BFD session parameters.

Procedure

1. Access global configuration mode.

   ```
   device# configure terminal
   ```

2. Access interface configuration mode.

   ```
   device(config)# interface ethernet 1/4
   ```

3. Specify the transmit time in milliseconds (ms), the receive time in ms, and the multiplier value that determines the number of consecutive control packets that must be missed before the connection to a peer is considered non-operational.

   ```
   device(conf-if-eth-1/4)# bfd interval 100 min-rx 100 multiplier 10
   ```

   This example specifies a transmit interval of 100 ms, the **min-rx** keyword followed by a receive interval of 100 ms, and the multiplier value of 10, which indicates that 10 consecutive packets must be missed before the connection to the peer is considered non-operational.

Example

This example configures BFD session parameters on a VE interface.

```
device# configure terminal
device(config)# interface ve 24
device(config-if-ve-24)# bfd interval 140 min-rx 125 multiplier 44
```

Disable BFD Sessions on an Interface

Procedure

1. Access global configuration mode.

```
device# configure terminal
```

2. Access interface configuration mode.

```
device(config)# interface ethernet 1/4
```

3. Disable BFD sessions on the interface.

```
device(config-if-eth-1/4)# bfd shutdown
```

BFD for BGP

Bidirectional Forwarding Detection (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 for BGP 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 single-hop and multihop iBGP and eBGP sessions with IPv4 or IPv6 neighbors in the default VRF and non-default VRF instances. BFD behavior is identical for iBGP and eBGP single-hop and multihop sessions and for IPv4 and IPv6 neighbors.

Consider the following when you configure BFD for BGP:

- Registration is global across all VRFs after 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. For single-hop sessions, the interface-level BFD transmit interval, receive interval, and detect multiplier are used.
- When a BGP session enters the Established state, BGP requests that BFD start a BFD session.
- BFD sessions are maintained across a BGP graceful restart.

BFD for BGP Session Creation and Deletion

The parameters for session creation are applied according to a specific hierarchy. When sessions are deleted, their states change according to the reason they were deleted.

For multihop BGP neighbors, session parameter values are applied according to the following hierarchy:

1. Values configured at the neighbor level
2. Values configured at the BGP neighbor group level
3. Values configured at the global BGP level
4. Default values

For single-hop neighbors, interface-level BFD parameters are applied.

BFD for BGP sessions are deleted in the following scenarios.

- **Session moves from Established to another state**. BFD sessions are deleted for a source or destination IPv4 or IPv6 address when a BGP session moves from the Established state to another BGP state.
- **A neighbor is unconfigured locally**. BFD sessions are deleted when a BGP session for a neighbor is unconfigured locally. If this unconfigured session 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 receiving this Cease message, the neighbor deletes the BFD session and moves to an Idle state.

Configure BFD Sessions for BGP

You can configure the intervals that a device waits to send (transmit) or receive control packets from Bidirectional Forwarding Detection (BFD) peers. You can also configure the interval after which routes are withdrawn when a BFD session is down.

Procedure

1. Access global configuration mode.
   ```
   device# configure terminal
   ```
2. Access BGP configuration mode.
   ```
   device(config)# router bgp
   ```
3. (Multihop sessions only) Specify the transmit time, the receive time, and the number of consecutive control packets that must be missed before the connection to a peer is considered non-operational.
   ```
   device(config-bgp-router)# bfd interval 140 min-rx 125 multiplier 44
   ```

   This example specifies a transmit interval of 140 ms, the **min-rx** keyword followed by a receive interval of 125 ms, and the multiplier value of 44, which indicates that 44 consecutive packets must be missed before the connection to the peer is considered non-operational.

   > **Note**
   > For single-hop sessions, run the **bfd interval** command at the interface level.

4. Specify the interval after which BGP routes are withdrawn after a BFD session is declared down.
   ```
   device(config-bgp-router)# bfd holdover-interval 15
   ```

   This example specifies a holdover interval of 15 seconds.

Enable BFD Sessions for a BGP Neighbor

You can configure the intervals that a device waits to send (transmit) or receive control packets from Bidirectional Forwarding Detection (BFD) peers. You can also configure the interval after which routes are withdrawn when a BFD session is down.

Procedure

1. Access global configuration mode.
   ```
   device# configure terminal
   ```
2. Access BGP configuration mode.
   ```
   device(config)# router bgp
   ```

3. To enable BFD sessions for a specified neighbor:

```
device(config-bgp-router)# neighbor 10.10.1.1 bfd
```

4. (Multihop sessions only) To specify the transmit time, the receive time, and the number of consecutive control packets that must be missed before the connection to a peer is considered non-operational:

```
device(config-bgp-router)# neighbor 10.10.1.1 bfd interval 120 min-rx 140 multiplier 10
```

For neighbor 10.10.1.1, this example specifies a transmit interval of 120 ms, the **min-rx** keyword followed by a receive interval of 140 ms, and the multiplier value of 10, which indicates that 10 consecutive packets must be missed before the connection to the peer is considered non-operational.

> **Note**
> For single-hop sessions, run the **bfd interval** command at the interface level.

5. To specify the interval (in seconds) after which BGP routes are withdrawn after a BFD session is declared down:

```
device(config-bgp-router)# neighbor 10.10.1.1 bfd holdover-interval 12
```

For neighbor 10.10.1.1, this example specifies a holdover interval of 12 seconds.

Enable BFD Sessions for a BGP Neighbor in a Non-default VRF

You can configure the intervals that a device waits to send (transmit) or receive control packets from Bidirectional Forwarding Detection (BFD) peers. You can also configure the interval after which routes are withdrawn when a BFD session is down.

Procedure

1. Access global configuration mode.

```
device# configure terminal
```

2. Access BGP configuration mode.

```
device(config)# router bgp
```

3. Access BGP address-family IPv4 unicast VRF configuration mode.

```
device(config-bgp-router)# address-family ipv4 unicast vrf green
```

4. To enable BFD sessions for a specified neighbor:

```
device(config-bgp-ipv4u-vrf)# neighbor 10.10.1.1 bfd
```

5. To specify the transmit time, the receive time, and the number of consecutive control packets that must be missed before the connection to a peer is considered non-operational:

```
device(config-bgp-ipv4u-vrf)# neighbor 10.10.1.1 bfd interval 120 min-rx 140
multiplier 10
```

For neighbor 10.10.1.1, this example specifies a transmit interval of 120 ms, the **min-rx** keyword followed by a receive interval of 140 ms, and the multiplier value of 10, which indicates that 10 consecutive packets must be missed before the connection to the peer is considered non-operational.

6. To specify the interval (in seconds) after which BGP routes are withdrawn after a BFD session is declared down:

```
device(config-bgp-ipv4u-vrf)# neighbor 10.10.1.1 bfd holdover-interval 12
```

For neighbor 10.10.1.1, this example specifies a holdover interval of 12 seconds.

Enable BFD Sessions for a BGP Peer Group

You can configure the intervals that a device waits to send (transmit) or receive control packets from Bidirectional Forwarding Detection (BFD) peers. You can also configure the interval after which routes are withdrawn when a BFD session is down.

Procedure

1. Access global configuration mode.

```
device# configure terminal
```

2. Access BCG configuration mode.

```
device(config)# router bgp
```

3. Create a peer group.

```
device(config-bgp-router)# neighbor pg1 peer-group
```

4. Enable BFD sessions for the peer group.

```
device(config-bgp-router)# neighbor pg1 bfd
```

5. To specify the transmit time, the receive time, and the number of consecutive control packets that must be missed before the connection to a peer is considered non-operational:

```
device(config-bgp-router)# neighbor pg1 bfd interval 200 min-rx 220 multiplier 25
```

For peer group 1, this example specifies a transmit interval of 200 ms, the **min-rx** keyword followed by a receive interval of 220 ms, and the multiplier value of 25, which indicates that 25 consecutive packets must be missed before the connection to the peer is considered non-operational.

6. To specify the interval after which BGP routes are withdrawn after a BFD session is declared down:

```
device(config-bgp-router)# neighbor pg1 bfd holdover-interval 17
```

For peer group 1, this example specifies a holdover interval of 17 seconds.

BFD for OSPF

Bidirectional Forwarding Detection (BFD) 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 support for OSPF is disabled by default.

Consider the following when you configure 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 the interface level and the 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

Sessions are created and deleted in various scenarios.

All Open Shortest Path First (OSPF) 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.

Each interface can have its own BFD timers. OSPF does not influence the BFD timer value for the session. The BFD module uses the configured BFD value or the default value 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.
> - OSPF neighbor sessions do not flap when BFD is enabled or disabled on the interface where the OSPF session is associated.
> - 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.

BFD for OSPF sessions are deleted in the following scenarios.

- **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 expiration and propagates this session down to OSPF to terminate the session.
- **OSPF is disabled on an interface**. All BFD sessions associated with each neighbor are deleted.
- **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.

Enable BFD on an OSPFv2-enabled Interface

BFD sessions can be enabled on one or more OSPFv2-enabled interfaces.

About This Task

This procedure is a prerequisite for Configure BFD for OSPFv2 Globally on page 119 and Configure BFD for OSPFv2 Globally in a Non-default VRF Instance on page 119.

Procedure

1. Access global configuration mode.

   ```
   device# configure terminal
   ```

2. Access interface configuration mode.

   ```
   device(config)# interface ethernet 1/4
   ```

3. Enable BFD on the interface.

   ```
   device(config-if-eth-1/4)# ip ospf bfd
   ```

Configure BFD for OSPFv2 Globally

BFD for OSPFv2 can be configured globally on interfaces where BFD is enabled.

Before You Begin

Enable BFD on OSPFv2 interfaces. For more information, see Enable BFD on an OSPFv2-enabled Interface on page 118.

Procedure

1. Access global configuration mode.

   ```
   device# configure terminal
   ```

2. Access OSPF router configuration mode.

   ```
   device(config)# router ospf
   ```

3. Enable BFD globally.

   ```
   device(config-router-ospf-vrf-default-vrf)# bfd
   ```

4. Specify the interval after which routes are withdrawn after a BFD session is declared down.

   ```
   device(config-router-ospf-vrf-default-vrf)# bfd holdover-interval 12
   ```

   This example specifies a holdover interval of 12 seconds.

Configure BFD for OSPFv2 Globally in a Non-default VRF Instance

In non-default VRF instances, BFD for OSPFv2 can be configured globally on interfaces where BFD is enabled.

Before You Begin

Enable BFD on OSPFv2 interfaces. For more information, see Enable BFD on an OSPFv2-enabled Interface on page 118.

Procedure

1. Access global configuration mode.

   ```
   device# configure terminal
   ```

2. Access OSPF router configuration mode and specify the non-default VRF instance.

   ```
   device(config)# router ospf vrf red
   ```

   This example accesses non-default VRF "red."

3. Enable BFD globally.

   ```
   device(config-router-ospf-vrf-red)# bfd
   ```

4. Specify the interval after which routes are withdrawn after a BFD session is declared down.

```
device(config-router-ospf-vrf-red)# bfd holdover-interval 12
```

This example specifies a holdover interval of 12 seconds.

Enable BFD on an OSPFv3-enabled Interface

BFD sessions can be configured on one or more OSPFv3-enabled interfaces.

About This Task

This procedure is a prerequisite for Configure BFD for OSPFv3 Globally on page 120 and Configure BFD for OSPFv2 Globally in a Non-default VRF Instance on page 119.

Procedure

1. Access global configuration mode.

```
device# configure terminal
```

2. Access interface configuration mode.

```
device(config)# interface ve 24
```

3. Enable BFD on the interface.

```
device(config-if-Ve-24)# ipv6 ospf bfd
```

Configure BFD for OSPFv3 Globally

BFD for OSPFv3 can be configured globally on interfaces where BFD is enabled.

Before You Begin

Enable BFD on OSPFv3 interfaces. For more information, see Enable BFD on an OSPFv3-enabled Interface on page 120

Procedure

1. Access global configuration mode.

```
device# configure terminal
```

2. Access OSPFv3 router configuration mode.

```
device(config)# ipv6 router ospf
```

3. Enable BFD globally.

```
device(config-ipv6-router-ospf-vrf-default-vrf)# bfd
```

4. Specify the interval after which routes are withdrawn after a BFD session is declared down.

```
device(config-ipv6-router-ospf-vrf-default-vrf)# bfd holdover-interval 20
```

This example specifies a holdover interval of 20 seconds.

Configure BFD for OSPFv3 Globally in a Non-default VRF Instance

In non-default VRF instances, BFD for OSPFv3 can be configured globally on interfaces where BFD is enabled.

Before You Begin

Enable BFD on OSPFv3 interfaces. For more information, see Enable BFD on an OSPFv3-enabled Interface on page 120.

Procedure

1. Access global configuration mode.

   ```
   device# configure terminal
   ```

2. Access ODPFv3 router configuration mode and specify the non-default VRF instance.

   ```
   device(config)# ipv6 router ospf vrf orange
   ```

   This example accesses non-default VRF "orange."

3. Enable BFD globally.

   ```
   device(config-ipv6-router-ospf-vrf-orange)# bfd
   ```

4. Specify the interval after which routes are withdrawn after a BFD session is declared down.

   ```
   device(config-ipv6-router-ospf-vrf-orange)# bfd holdover-interval 22
   ```

   This example specifies a holdover interval of 22 seconds.

Display BFD Information

You can use **show** commands to display information about BFD sessions and neighbors.

Procedure

1. Display global BFD information.

   ```
   device# show bfd
     BFD State: ENABLED, Version: 1
     Supported Protocols: ospf6, ospf
     All Sessions: Current: 2 Max Allowed: 250 Max Exceeded Count: 0

     Port Sessions     MinTx        MinRx        Mult Sessions
     ====              =====        =====        ==== ========
     Eth 0/1           300          300          3    2
   ```

2. Display BFD neighbor information for the default VRF.

   ```
   device# show bfd neighbors

   Flags: * indicates State is inconsistent across the cluster
   OurAddr      NeighAddr    State      Int
   =======      =========    =====      ===
   32.1.1.2     32.1.1.1     UP         Eth 0/7
   Local    State: UP          Diag: 0            Demand mode: 0    Poll: 0
   Received State: UP          Diag: 0            Demand mode: 0    Poll: 0    Final:
   0
   Local    MinTxInt(ms): 300    MinRxInt(ms): 300    Multiplier: 3
   Received MinTxInt(ms): 300    MinRxInt(ms): 300    Multiplier: 3
   Rx Count: 59                Tx Count: 49
   LD/RD:          1/1         Heard from Remote: Y
   Current Registered Protocols: ospf
   Uptime: 0 day 3 hour 4 min 2 sec 650 msec
   ```

3. Display BFD neighbor information with OSPF details.

```
device# show bfd neighbors application ospf details

Flags: * indicates State is inconsistent across the cluster
OurAddr      NeighAddr    State     Int
=======      =========    =====     ===
7.7.7.1      7.7.7.2      UP        Eth 0/1/2

Local    State: UP  Diag: 0  Demand mode: 0 Poll: 0
Received State: UP  Diag: 0  Demand mode: 0 Poll: 0 Final: 0
Local    MinTxInt(ms): 1000   MinRxInt(ms): 1000   Multiplier: 5
Received MinTxInt(ms): 1000   MinRxInt(ms): 1000   Multiplier: 5
  Rx Count: 3806   Tx Count: 4308
  LD/RD: 10001/10001   Heard from Remote: Y
Current Registered Protocols: ospf
Uptime: 0 day 0 hour 0 min 0 sec 0 msec
```

4. Display BFD neighbor information with details about the destination device.

```
device# show bfd neighbors dest-ip 1.1.1.6 details
Flags: * indicates State is inconsistent across the cluster
OurAddr                               NeighAddr
State     Int
=======                               =========
=====     ===
1.1.1.1
1.1.1.6                                          UP         Eth 0/1

Local    State: UP           Diag: 0            Demand mode: 0    Poll: 0
Received State: UP           Diag: 0            Demand mode: 0    Poll: 0    Final:
0
Local    MinTxInt(ms): 300   MinRxInt(ms): 300   Multiplier: 3
Received MinTxInt(ms): 300   MinRxInt(ms): 300   Multiplier: 3
Rx Count: 0                  Tx Count: 0
LD/RD:         5/4           Heard from Remote: Y
Current Registered Protocols: ospf
Uptime: 0 day 16 hour 29 min 1 sec 90 msec
```

5. Display BFD neighbor information with details about a specific VE interface.

```
device# show bfd neighbors interface ve 5 details
Flags: * indicates State is inconsistent across the cluster
OurAddr                               NeighAddr                    State     Int
=======                               =========                    =====     ===
124.1.1.2                             124.1.1.1                     UP
Ve5

Local    State: UP           Diag: 0            Demand mode: 0    Poll: 0
Received State: UP           Diag: 0            Demand mode: 0    Poll: 0    Final:
0
Local    MinTxInt(ms): 300   MinRxInt(ms): 300   Multiplier: 3
Received MinTxInt(ms): 300   MinRxInt(ms): 300   Multiplier: 3
Rx Count: 0                  Tx Count: 0
LD/RD:         5/4           Heard from Remote: Y
Current Registered Protocols: ospf
Uptime: 0 day 16 hour 31 min 41 sec 421 msec
```

BGP4

BGP4 Overview

Border Gateway Protocol version 4 (BGP4) is an exterior gateway protocol that performs inter-autonomous system and inter-domain routing.

BGP4 peers to other BGP-speaking systems over TCP to exchange network reachability and routing information. BGP primarily performs two types of routing.

- Inter-autonomous system routing, called Exterior BGP (eBGP)
- Intra-autonomous system routing, called Interior BGP (iBGP)

BGP peers in different autonomous systems (AS) use eBGP. Within an AS, BGP can be used to maintain a consistent view of network topology, to provide optimal routing, or to scale the network.

Devices in the same AS use different Interior Gateway Protocols (IGP), such as IS-IS and OSPF, to communicate with one another. In an AS, BGP can be used to maintain a consistent view of network topology, to provide optimal routing, and to scale the network.

As a path vector protocol, BGP treats each AS as one point on the path to a given destination. For each route (destination), BGP maintains the AS path and uses this information to detect and prevent loops between autonomous systems.

BGP4 is the standard Exterior Gateway Protocol (EGP) used by Internet devices and is the EGP implemented on SLX-OS devices.

The following diagram is a simple example of two BGP4 autonomous systems, AS1 and AS2. Each system contains three BGP4 devices. To communication in their system, devices use iBGP. To communicate to the other system, devices use eBGP. Notice that each device also runs an IGP. The devices in AS1 run OSPF and the devices in AS2 run IS-IS. The devices can be configured to redistribute routes among BGP4, IS-IS, and OSPF. They also can redistribute static routes.
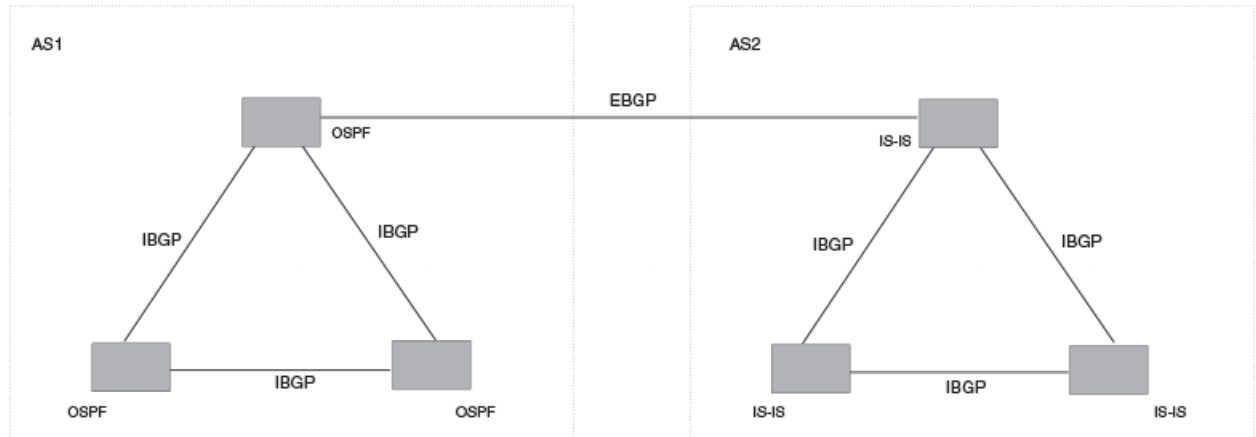
Figure 12: BGP4 autonomous systems

BGP4 Peering

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 exchanges routing information with another speaker (in the same or a different autonomous system) by using a TCP connection to port 179 (the well-known BGP port). 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

A BGP4 session between peers in the same autonomous system is an Interior BGP (iBGP) session. A session between peers in different autonomous systems is an Exterior BGP (eBGP) session.

To establish a TCP connection between two iBGP peers, IP reachability should be established by means of the underlying IGP protocol (for example, OSPF) or by means of static routes. When routes are advertised in iBGP peers, the following actions do not usually occur:

- Routes learned from an iBGP peer are not usually advertised to other iBGP peers, to prevent loops in an autonomous system.
- Path attributes are not usually changed, to maintain the best path selection at other nodes in an autonomous system.
- The autonomous system path and next hop are not usually changed.

BGP4 Message Types

BGP4 messages can be of the following types: OPEN, UPDATE, NOTIFICATION, KEEPALIVE, or ROUTE-REFRESH.

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.

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 autonomous system numbers are supported.

KEEPALIVE and HOLDTIME timers

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-times, the lowest value is used. A hold-time of 0 means "always consider neighbor to be active."

BGP Identifier

Indicates the router (or device) ID of the sender. When a router-id is not configured, the device-id is taken from the loopback interface. Otherwise, the lowest IP address in the system is used.

Parameter List

An optional list of additional parameters used in peer negotiation.

UPDATE message

The UPDATE message advertises new routes, withdraws previously advertised routes, or both. The UPDATE message passes BGP4 attributes to describe the characteristics of a BGP path by the advertising device.

Withdrawn Routes Length	Withdrawn Routes	Total Path Attributes Len	Path Attributes	NLRI
2	variable	2	variable	variable

Withdrawn Routes Length

Indicates the length of the next (withdrawn routes) field. It can be 0.

Withdrawn Routes

Contains a list of routes (or IP-prefix/Length) to indicate routes being withdrawn.

Total Path Attribute Len

Indicates the length of the 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. All well-known attributes, as described in RFC 4271, are supported.

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

If an error causes the TCP connection to close, the closing peer sends a notification message to indicate the type of error.

Error Code	Error Subcode	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 reported error.

Error Data

Provides data based on the error code and the subcode.

KEEPALIVE message

Because BGP does not regularly exchanges route updates to maintain a session, KEEPALIVE messages are sent to keep the session alive. The KEEPALIVE time specifies how frequently the device sends KEEPALIVE messages to its BGP4 neighbors.

A KEEPALIVE message contains the BGP header without a data field. The 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 message is useful when the inbound policy has been changed.

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).

BGP4 Best Path Selection Algorithm

BGP routers receive multiple paths to the same destination. The algorithm determines the best path.

About This Task

The BGP decision process is applied to the routes 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 stored in the Adjacency RIB, Outgoing (RIB-Out).

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.

Procedure

1. Verify that the next hop can be resolved by means of the 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.

   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 only if the routes were learned from the same neighboring autonomous system. 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 autonomous system 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 in the local autonomous system or confederation, unless the **compare-med-empty-aspath** command is configured.

8. Prefer paths in the following order:

   - Routes received through EGP from a BGP4 neighbor outside of the confederation
   - Routes received through EGP from a BGP4 device in the confederation *or* routes received through IGP.

9. If all the comparisons are equal, prefer the route with the lowest IGP metric to the BGP4 next hop. This is the closest internal path in the autonomous system to reach the destination.

10. If the internal paths are the same and BGP4 load sharing is enabled, load share among the paths. Otherwise go to Step 11.

    > **Note**
    > For EGP routes, load sharing applies only when the paths are from neighbors in the same remote autonomous system. EGP paths from neighbors in different autonomous systems 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.

BGP Device ID

BGP automatically calculates the device identifier that it uses to specify the origin in routes that it advertises.

If a router-id configuration is present in the system, then device-id is used as the router-id. Otherwise, the device checks for a loopback interface. The IP address configured on that interface is chosen as the device-id.

If a loopback interface is not configured, the device-id is chosen from the lowest-numbered IP interface address configured on the device. After the device-id is chosen, the device identifier is not calculated unless the configured IP address is deleted.

NEW! Enable BGP4 Globally

When BGP4 is enabled, you can configure more advanced features such as route dampening and autonomous system confederation.

Procedure

1. Access global configuration mode.

```
device# configure terminal
```

2. Enable BGP4 globally.

```
device(config)# router bgp
```

Example

Several configurations are available from BGP router 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
  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
  describe                   Display transparent command information
  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 asleast
                             desirable
  neighbor                   Specify a neighbor router

  timers                     Adjust routing timers
```

Enable 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. From there, you can access several commands that allow the configuration of BGP4 for the non-default VRF instance.

Before You Begin
Before performing this task, ensure that a non-default VRF instance is configured.

Procedure

1. Access global configuration mode.

   ```
   device# configure terminal
   ```

2. Access BGP configuration mode.

   ```
   device(config)# router bgp
   ```

3. Specify the VRF name and enter BGP address-family IPv4 unicast VRF configuration mode.

   ```
   device(config-bgp-router)# address-family ipv4 unicast vrf red
   device(config-bgp-ipv4u-vrf)#
   ```

   This example shows the **address-family ipv4 unicast** command, the **vrf** keyword, and the VRF name (red).

Example

The following example summarizes the commands in this procedure.

```
device# configure terminal
device(config)# router bgp
device(config-bgp-router)# address-family ipv4 unicast vrf red
device(config-bgp-ipv4u-vrf)#
```

Configure a Local AS Number for a BGP4 Device

The local autonomous system number (ASN) identifies the autonomous system in which the BGP device resides.

About This Task

If the ASN of the organization is unknown, use a well-known private ASN in the range from 64512 through 65535.

Procedure

1. Access global configuration mode.

   ```
   device# configure terminal
   ```

2. Access BGP router configuration mode.

   ```
   device(config)# router bgp
   ```

3. Specify the ASN for the autonomous system in which your device resides.

   ```
   device(config-bgp-router)# local-as 1000
   ```

Example

The following example summarizes the commands in this procedure.

```
device# configure terminal
device(config)# router bgp
device(config-bgp-router)# local-as 1000
```

BGP4 and Address-Family IPv4 Unicast Mode

You can configure BGP4 unicast routes from the address-family IPv4 unicast configuration level.

You can access the address-family IPv4 unicast configuration level from BGP configuration mode.

```
device(config)# router bgp
device(config-bgp-router)# address-family ipv4 unicast
device(config-bgp-ipv4u)#
```

> **Note**
>
> The commands that are available from the address-family IPv4 unicast configuration level are also available from the address-family IPv6 unicast configuration level. Each configuration level allows you to access commands that apply only to that particular address family.

Several configuration options are available from 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-mpls                 Changes for IPoMPLS route download, pkt path.
  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
```

BGP4 Neighbor Configuration

For each neighbor that a device peers with, a neighbor configuration specifies an IP address and the ASN of the neighbor.

The IP address of the neighbor is the primary IP address of the interface connection. For each neighbor, you can specify a set of attributes. In cases where a set of neighbors share the same set of attributes, it is a best practice to create a peer-group. For more information, see BGP4 Peer Groups on page 135.

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 are available from BGP global configuration

mode. The parameters that are specific to an address family are available from BGP address-family submode.

Several neighbor configuration options are available from BGP global configuration mode.

```
device(config-bgp-router)# neighbor 10.1.1.1 ?
Possible completions:
  advertisement-interval   Minimum interval between sending BGP routing updates
  as-override              Override matching AS-number while sending update
  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               Assign peer-group to neighbor
  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
```

Several neighbor configuration options are available from BGP IPv4 address family unicast configuration mode.

```
device(config-bgp-ipv4u)# 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
  capability               Advertise capability to the peer
  default-originate        Originate default route to peer
  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
```

Configure a BGP4 Neighbor

You can configure the ASN of the autonomous system in which the remote neighbor resides.

Procedure

1. Access global configuration mode.

   ```
   device# configure terminal
   ```

2. Access BGP configuration mode.

   ```
   device(config)# router bgp
   ```

3. Specify the autonomous system in which your device resides.

```
device(config-bgp-router)# local-as 1000
```

4. Specify the ASN for the autonomous system in which the neighbor resides.

```
device(config-bgp-router)# neighbor 10.1.1.1 remote-as 1001
```

This example configures ASN 1001 for the neighbor with IP address 10.1.1.1.

Example

This example summarizes the commands in this procedure.

```
device# configure terminal
device(config)# router bgp
device(config-bgp-router)# local-as 1000
device(config-bgp-router)# neighbor 10.1.1.1 remote-as 1001
```

BGP4 Peer Groups

You can use the **neighbor peer-group** command to group IPv4 neighbors that have the same attributes and parameters.

You can associate neighbor IPv4 addresses with a 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 for a peer group.

- Reset neighbor sessions
- Perform soft-outbound resets, in which 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 for a neighbor takes precedence over the attribute value configured for a peer group. The default attribute is used if the neither the peer group nor the neighbor has the attribute configured.

For the parameters of a BGP4 peer group to take effect, you must activate the peer group in IPv4 unicast address family configuration mode.

Create a BGP4 Peer Group

A BGP4 peer group is a collection of IPv4 neighbors that have the same attributes and parameters.

Procedure

1. Access global configuration mode.

```
device# configure terminal
```

2. Access BGP configuration mode.

```
device(config)# router bgp
```

3. Specify the autonomous system in which your device resides.

```
device(config-bgp-router)# local-as 1000
```

4. Create a peer group.

```
device(config-bgp-router)# neighbor mypeergroup1 peer-group
```

This example shows the **neighbor** command, the peer group name (mypeergroup1), and the **peer-group** keyword.

5. Specify the autonomous system number (ASN) of the peer group.

```
device(config-bgp-router)# neighbor mypeergroup1 remote-as 11
```

This example associates ASN 11 to the new peer group.

6. Associate a neighbor with the peer group.

```
device(config-bgp-router)# neighbor 10.2.2.2 peer-group mypeergroup1
```

This example shows the **neighbor** command, the IP address of the neighbor, the **peer-group** keyword, and the name of the peer group (mypeergroup1).

7. Repeat step 6 as needed to associate more neighbors with the peer group.

Example

The following example summarizes the commands in this procedure

```
device# configure terminal
device(config)# 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
```

Four-byte AS Numbers for BGP4

When four-byte autonomous system number (ASN) functionality is enabled, a device announces and negotiates "AS4" capability with its neighbors.

You can enable AS4 capability at the BGP global level or at the neighbor or peer group level. The following combinations are supported.

- AS4 capability enabled for a neighbor or peer group but disabled at the global level
- AS4 capability enabled at the global level but disabled for a neighbor or peer group

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 not configured at the neighbor level and the peer group level.

Peering is not established when a device with a 4-byte ASN tries to connect to a device that does not have AS4 support.

Enable ASN Capability Globally for BGP4

You can configure 4-byte autonomous system number (ASN) capability at the BGP4 global level.

Procedure

1. Access global configuration mode.

   ```
   device# configure terminal
   ```

2. Access BGP configuration mode.

   ```
   device(config)# router bgp
   ```

3. Enable 4-byte ASN capability.

   ```
   device(config-bgp-router)# capability as4-enable
   ```

Enable ASN Capability for a BGP4 Neighbor

You can configure 4-byte autonomous system number (ASN) capability for a neighbor or peer-group.

Procedure

1. Access global configuration mode.

   ```
   device# configure terminal
   ```

2. Access BGP configuration mode.

   ```
   device(config)# router bgp
   ```

3. Specify the autonomous system in which your device resides.

   ```
   device(config-bgp-router)# local-as 1000
   ```

4. Enable 4-byte ASN capability for a neighbor.

   ```
   device(config-bgp-router)# neighbor 10.1.2.3 capability as4 enable
   ```

   This example shows the **neighbor** command, the IP address of the neighbor, and the **capability as4** keywords.

Example

The following example summarizes the commands in this procedure.

```
device# configure terminal
device(config)# router bgp
device(config-bgp-router)# local-as 1000
device(config-bgp-router)# neighbor 10.1.2.3 capability as4 enable
```

Redistribute Routes into BGP4

You can advertise routes into BGP4 by redistributing static, connected, ISIS, or OSPF routes.

About This Task

Routes learned through BGP4 can be redistributed into OSPF or IS-IS. You can specify an optional route-map, which is consulted before routes are added to BGP. Management routes are not redistributed.

Procedure

1. Access global configuration mode.

```
device# configure terminal
```

2. Access BGP configuration mode.

```
device(config)# router bgp
```

3. Access BGP IPv4 address family unicast configuration mode.

```
device(config-bgp-router)# address-family ipv4 unicast
```

4. Redistribute connected routes.

```
device(config-bgp-ipv4u)# redistribute connected
```

Example

This example redistributes IS-IS routes with level 1 packets.

```
device# configure terminal
device(config)# router bgp
device(config-bgp-router)# address-family ipv4 unicast
device(config-bgp-ipv4u)# redistribute isis level-1
```

Example

This example redistributes static routes with a metric of 200.

```
device# configure terminal
device(config)# router bgp
device(config-bgp-router)# address-family ipv4 unicast
device(config-bgp-ipv4u)# redistribute static metric 200
```

Example

This example redistributes OSPF external type 1 routes with a metric of 200.

```
device# configure terminal
device(config)# router bgp
device(config-bgp-router)# address-family ipv4 unicast
device(config-bgp-ipv4u)# redistribute ospf match external1 metric 200
```

Import Routes into BGP4

Routes can be explicitly imported into BGP4 for advertisement.

About This Task

You can specify a route to be local. If the same route is received by means of EGP, the local IGP route is 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.

This procedure imports a route with a routemap. For more examples, see the *Extreme SLX-OS Command Reference*.

Before You Begin
With the exception of static network routes, the routes that are imported into BGP4 must first exist in the IPv4 unicast route table.

Procedure

1. Access global configuration mode.

   ```
   device# configure terminal
   ```

2. Access BGP configuration mode.

   ```
   device(config)# router bgp
   ```

3. Access BGP address-family IPv4 unicast configuration mode.

   ```
   device(config-bgp-router)# address-family ipv4 unicast
   ```

4. Import the specified prefix into the BGP4 database.

   ```
   device(config-bgp-ipv4u)# network 10.1.1.1/32 route-map myroutemap
   ```

   This example imports the IP prefix 10.1.1.1/32 into the BGP4 database and specifies a route map called myroutemap.

Static BGP4 Networks

Before advertising a route, BGP checks for its existence in the routing table. You can configure BGP to advertise a stable route, which does not need to exist in the routing table.

By configuring a static BGP4 network, you creating a stable network in the core. Although a static route does not flap unless it is manually deleted, a static BGP4 network does not interrupt the normal BGP4 decision process on other learned routes that are installed into the Routing Table Manager (RTM). Consequently, any route that can be resolved is installed into the RTM.

When the configured route is lost, BGP installs the "null0" route in the routing table. The null0 route is removed when the route is resolved.

Configure a Static BGP4 Network

BGP can advertise a stable route that does not need to exist in routing table.

About This Task
This task configures a static network and sets the administrative distance.

Procedure

1. Access global configuration mode.

   ```
   device# configure terminal
   ```

2. Access BGP configuration mode.

   ```
   device(config)# router bgp
   ```

3. Specify the autonomous system in which your device resides.

   ```
   device(config-bgp-router)# local-as 1000
   ```

4. Access IPv4 address family configuration mode.

   ```
   device(config-bgp-router)# address-family ipv4 unicast
   ```

5. Configure a static network and set an administrative distance.

   ```
   device(config-bgp-ipv4u)# static-network 10.11.12.0/32 distance 300
   ```

   This example configures 10.11.12.0/32 as a static network and sets an administrative distance of 300.

Example

The following example summarizes the commands in this procedure.

```
device# configure terminal
device(config)# 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
```

BGP4 Route Reflection

Route-reflector clients advertise routes to the route reflector. The route reflector, in turn, reflects the advertisements to other route-reflector clients.

A BGP device can act as a route-reflector client or as a route reflector. Run the **neighbor route-reflector-client** command on a host device to configure a neighbor to be a route-reflector client. The host then becomes as the route reflector.

Multiple route reflectors should 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 reflects routes as follows:

- Routes from route-reflector clients are reflected to other client peers and to non-client peers.
- Routes from non-client 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 routes that are preferred and installed into the Routing Table Manager (RTM). The route reflector may not reflect routes that are not installed into the RTM because the routing table is full. You can use the **always-propagate** command to configure the route reflector to reflect routes that are not in the RTM.

Configure a Cluster ID for a BGP4 Route Reflector

When you have more than one route reflector, change the cluster ID so that all route reflectors belong to the same cluster.

About This Task

Multiple route reflectors should belong to the same cluster. By default, the value for the **cluster-id** command is the device ID. Use this procedure to change the device ID.

Procedure

1. Access global configuration mode.

   ```
   device# configure terminal
   ```

2. Access BGP configuration mode.

   ```
   device(config)# router bgp
   ```

3. Specify the autonomous system in which your device resides.

   ```
   device(config-bgp-router)# local-as 1000
   ```

4. Change the device ID.

```
device(config-bgp-router)# cluster-id 321
```

This example changes the cluster ID of a device from the default value to 321.

Example

The following example summarizes the commands in this procedure.

```
device# configure terminal
device(config)# router bgp
device(config-bgp-router)# local-as 1000
device(config-bgp-router)# cluster-id 321
```

Configure a BGP4 Route-Reflector Client

You can configure a BGP peer as a route-reflector client.

Procedure

1. Access global configuration mode.

```
device# configure terminal
```

2. Access BGP configuration mode.

```
device(config)# router bgp
```

3. Specify the autonomous system in which your device resides.

```
device(config-bgp-router)# local-as 1000
```

4. Access IPv4 address family configuration mode.

```
device(config-bgp-router)# address-family ipv4 unicast
```

5. Specify the neighbor to be a route-reflector client.

```
device(config-bgp-ipv4u)# neighbor 10.1.1.1 route-reflector-client
```

This example configures neighbor 10.1.1.1 as a route-reflector client.

Example

This example summarizes the commands in this procedure.

```
device# configure terminal
device(config)# 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
```

Aggregate the Routes to Advertise to BGP4 Neighbors

By default, a device advertises individual routes for all networks. To minimize the size of the routing table, you can aggregate the routes in a range of networks into one IPv4 prefix.

About This Task

You can aggregate BGP4 routes into one network prefix, for example, 10.1.1.1/32. This single prefix is advertised to BGP4 neighbors instead of the individual BGP routes in the routing table that are within the specified range.

Procedure

1. Access global configuration mode.

   ```
   device# configure terminal
   ```

2. Access BGP configuration mode.

   ```
   device(config)# router bgp
   ```

3. Access BGP address family IPv4 unicast configuration mode.

   ```
   device(config-bgp-router)# address-family ipv4 unicast
   ```

4. Specify the network prefix into which routes are to be aggregated.

   ```
   device(config-bgp-ipv4u)# aggregate-address 10.1.1.1/32
   ```

Example

The following example summarizes the commands in this procedure.

```
device# configure terminal
device(config)# router bgp
device(config-bgp-router)# address-family ipv4 unicast
device(config-bgp-ipv4u)# aggregate-address 10.1.1.1/32
```

Advertise the Default BGP4 Route

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.

About This Task

By default, a BGP4 device does not originate and advertise the default BGP4 route. You can configure a BGP4 device to advertise the default IPv4 route to all BGP4 neighbors and to install that route in the local BGP4 route table.

Before You Begin

The default route must be present in the local IPv4 route table.

Procedure

1. Access global configuration mode.

   ```
   device# configure terminal
   ```

2. Access BGP configuration mode.

   ```
   device(config)# router bgp
   ```

3. Access IPv4 address family unicast configuration mode.

   ```
   device(config-bgp-router)# address-family ipv4 unicast
   ```

4. Configure the device to originate and advertise the default BGP route and to install that route in the local BGP4 route table.

   ```
   device(config-bgp-ipv4u)# default-information-originate
   ```

Example

This example summarizes the commands in this procedure.

```
device# configure terminal
device(config)# router bgp
```

```
device(config-bgp-router)# address-family ipv4 unicast
device(config-bgp-ipv4u)# default-information-originate
```

Advertise the Default BGP4 Route to a Specific Neighbor

You can configure a BGP device to advertise the default IPv4 route to a specific neighbor.

Procedure

1. Access global configuration mode.

   ```
   device# configure terminal
   ```

2. Access BGP configuration mode.

   ```
   device(config)# router bgp
   ```

3. Specify the autonomous system in which your device resides.

   ```
   device(config-bgp-router)# local-as 1000
   ```

4. Access address-family IPv4 unicast configuration mode.

   ```
   device(config-bgp-router)# address-family ipv4 unicast
   ```

5. 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
   ```

   This example enables the device to advertise the default route to the 10.4.4.4 neighbor.

Example
This example summarizes the commands in this procedure.

```
device# configure terminal
device(config)# 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
```

Use the Default BGP4 Route as a Valid Next Hop

You can configure a device to use the default IPv4 route as a valid next hop for a BGP4 route. This configuration is helpful in certain cases, such as when a device is acting as an edge device.

About This Task
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.

Procedure

1. Access global configuration mode.

   ```
   device# configure terminal
   ```

2. Access BGP configuration mode.

   ```
   device(config)# router bgp
   ```

3. Access address-family IPv4 unicast configuration mode.

   ```
   device(config-bgp-router)# address-family ipv4 unicast
   ```

4. Configure the device to use the default route as a valid next hop.

```
device(config-bgp-ipv4u)# next-hop-enable-default
```

BGP4 Multipath for Load Balancing

By default, BGP balances traffic across one path. You can use the multipath feature to enable load-balancing across different BGP4 paths.

When BGP4 multipath load sharing is enabled, iBGP and eBGP paths are eligible for load sharing. 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.

Enable Load Balancing Across Different BGP4 Paths

You can use the BGP4 multipath feature to enable load balancing different BGP4+ paths.

About This Task

You can use the *num* variable to specify a maximum number of shared paths. Or you can use the **use-load-sharing** keyword to use the value that has already configured by means of the **ip load-sharing** command.

Procedure

1. Access global configuration mode.

```
device# configure terminal
```

2. Access BGP configuration mode.

```
device(config)# router bgp
```

3. Access IPv4 address family configuration mode.

```
device(config-bgp-router)# address-family ipv4 unicast
```

4. Perform one of the following steps to specify load-sharing paths:

   - Use the *num* variable to specify a maximum number of shared paths.
     ```
     device(config-bgp-ipv4u)# maximum-paths 8
     ```

   - Use the value that has been configured by means of the **ip load-sharing** command.
     ```
     device(config-bgp-ipv4u)# maximum-paths use-load-sharing
     ```

Example

The following example summarizes the commands in this procedure

```
device# configure terminal
device(config)# router bgp
device(config-bgp-router)# address-family ipv4 unicast
device(config-bgp-ipv4u)# maximum-paths 8
device(config-bgp-ipv4u)# maximum-paths use-load-sharing
```

Change the Weight Added to Received BGP4 Routes

The device adds weight to routes that are received from BGP neighbors. In the BGP4 path-selection algorithm, routes with larger weights have preference over routes with smaller weights.

About This Task

You can change the weight of routes that are received from a specified BGP neighbor.

Procedure

1. Access global configuration mode.

   ```
   device# configure terminal
   ```

2. Access BGP configuration mode.

   ```
   device(config)# router bgp
   ```

3. Specify the autonomous system in which your device resides.

   ```
   device(config-bgp-router)# local-as 1000
   ```

4. Access address-family IPv4 unicast configuration mode.

   ```
   device(config-bgp-router)# address-family ipv4 unicast
   ```

5. Specify the neighbor and the weight that you want to add to routes that are received from that neighbor.

   ```
   device(config-bgp-ipv4u)# neighbor 10.11.12.13 weight 100
   ```

   This example shows the **neighbor** command, the IP address of the neighbor, the **weight** keyword, and the weight value of 100.

Example

This example summarizes the commands in this procedure.

```
device# configure terminal
device(config)# router bgp
device(config-bgp-router)# local-as 1000
device(config-bgp-router)# address-family ipv4 unicast
device(config-bgp-ipv4u)# neighbor 10.11.12.13 weight 100
```

Next-hop Recursion for BGP4

A device performs a route lookup for learned BGP4 routes to find the IPv4 address of the next hop. If the BGP4 next hop is not the immediate next hop, a second lookup (a recursive lookup) is required to resolve the exit path for destination traffic.

A BGP4 route is eligible for addition to the IPv4 route table if the following conditions are true:

- The lookup obtains 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, a device performs only one lookup for the next-hop IPv4 address. If the lookup does not find a valid next hop IPv4 address, or if 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 situation can occur when the IGPs do not learn a complete set of IGP routes. Instead, 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 finds an iBGP path that originated in the same autonomous system, then the next hop is considered resolved and BGP4-dependent routes are eligible for addition in the IPv4 route table.

Enable Next-hop Recursion for BGP4

With next-hop recursion enabled, a device can find the IGP route to the next hop for a BGP4 route.

Procedure

1. Access global configuration mode.

```
device# configure terminal
```

2. Access BGP configuration mode.

```
device(config)# router bgp
```

3. Access address-family IPv4 unicast configuration mode.

```
device(config-bgp-router)# address-family ipv4 unicast
```

4. Enable recursive next-hop lookups.

```
device(config-bgp-ipv4u)# next-hop-recursion
```

BGP4 Route Filters

You use route filters to select and identify routes that are advertised or received from neighbor routers.

BGP4 supports several route filters.

AS-path filter

An AS-path filter allows or denies specified prefixes from a specified autonomous system (AS). You use regular expression to match the prefixes and AS.

Community filter

A BGP community is a group of destinations that share a common property. You can filter for communities of a specified type.

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.

Prefix list

A prefix list consists of one or more conditional statements that pose an action (permit or deny) if a route matches a specified prefix.

Route map

A route map is a set of match conditions and parameter settings that the device uses to change route attributes and to control the redistribution of routes into other protocols.

Table map

A table map maps external entry attributes into the BGP routing table to preserve those attributes after they are redistributed into OSPF. For more information, see the **table-map** command in the *Extreme SLX-OS Command Reference*.

BGP4 Regular Expression Pattern-matching

The following table illustrates the functions of BGP regular expression pattern-matching characters and illustrates their use.

Table 19: 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.
?	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: for example, b8 and W4.

Table 19: BGP regular expression pattern-matching characters (continued)

Regular expression character	Function	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, .

BGP4 Outbound Route Filtering

You use the BGP4 Outbound Route Filtering (ORF) feature to minimize the number of BGP updates sent between BGP peers.

When ORF is enabled, unwanted routing updates are removed, 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. The remote peer applies the filter as an outbound filter for the neighbor.

You can configure ORF 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. Using send mode, the local peer exchanges the ORF capability with a remote peer for a prefix list that is configured as an inbound filter for that peer locally.

The remote peer sends the first update after it receives a ROUTE REFRESH request or BGP ORF with IMMEDIATE from the peer. The local and remote peers exchange updates to maintain the ORF on each router.

Configure BGP4 Outbound Route Filtering

You can configure BGP4 Outbound Route Filtering (ORF) in receive mode, send mode, or both, minimizing the number of BGP updates exchanged between BGP peers.

Procedure

1. Access global configuration mode.

   ```
   device# configure terminal
   ```

2. Access BGP configuration mode.

   ```
   device(config)# router bgp
   ```

3. Access IPv4 address family configuration mode.

   ```
   device(config-bgp-router)# address-family ipv4 unicast
   ```

4. (For send mode only) Filter the incoming route updates from a specified BGP neighbor.

```
device(config-bgp-ipv4u)# neighbor 10.1.2.3 prefix-list myprefixlist in
```

This example specifies the **neighbor prefix-list** command, the IPv4 address of the neighbor, a prefix list named myprefixlist, and the keyword **in**.

5. To advertise ORF send capabilities, run the **neighbor capability orf prefixlist** command with the **send** keyword.

```
device(config-bgp-ipv4u)# neighbor 10.1.2.3 capability orf prefixlist send
```

6. To advertise ORF receive capabilities, run the **neighbor capability orf prefixlist** command with the **receive** keyword.

```
device(config-bgp-ipv4u)# neighbor 10.1.2.3 capability orf prefixlist receive
```

7. To advertise both ORF send and receive capabilities, run the **neighbor capability orf prefixlist** command.

```
device(config-bgp-ipv4u)# neighbor 10.1.2.3 capability orf prefixlist
```

Example

The following example summarizes the commands for advertising ORF in receive mode.

```
device# configure terminal
device(config)# router bgp
device(config-bgp-router)# address-family ipv4 unicast
device(config-bgp-ipv4u)# neighbor 10.1.2.3 capability orf prefixlist receive
```

Example

The following example summarizes the commands for advertising ORF in send mode.

```
device# configure terminal
device(config)# 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
```

Example

The following example summarizes the commands for advertising ORF in both send and receive modes.

```
device# configure terminal
device(config)# 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
```

Cooperative BGP4 Route Filtering

Cooperative filtering conserves resources by eliminating unnecessary route updates and filter processing.

By default, the device filters incoming routes locally, on the device itself. With cooperative BGP4 route filtering, the neighbor performs the filtering before sending the routes to the device. For example, the device sends 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 use to generate the route updates, and the device saves the resources it would use to filter the routes.

When you enable cooperative filtering, the device advertises this capability in its OPEN message to the neighbor. 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.

The following is the high-level process for configuring cooperative filtering on the device and on the BGP4 neighbor.

1. Configure the filter. Cooperative filtering is currently supported only for filters configured using IP prefix lists.
2. Apply the filter as an inbound filter to the neighbor.
3. 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 it receives 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.
4. Reset the BGP4 neighbor session to send and receive ORFs.
5. 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.

Enable BGP4 Cooperative Route Filtering

You can use cooperative BGP4 route filtering to cause the neighbor to perform the filtering before sending the routes to the device.

About This Task
Cooperative filtering conserves resources by eliminating unnecessary route updates and filter processing.

Procedure

1. Access global configuration mode.

   ```
   device# configure terminal
   ```

2. Configure the IP prefix list instance.

   ```
   device(config)# ip prefix-list Routesfrom1234 deny 10.20.0.0/24
   device(config)# ip prefix-list Routesfrom1234 permit 10.0.0.0/0 le 32
   ```

   The first example denies the routes in the Routesfrom1234 prefix list for the specified IP prefix and prefix length. The second example permits the routes in the Routesfrom1234 prefix list for the specified IP prefix and prefix length, where the subnet mask length must be less than or equal to 32.

3. Access BGP configuration mode.

   ```
   device(config)# router bgp
   ```

4. Access IPv4 address family configuration mode.

   ```
   device(config-bgp-router)# address-family ipv4 unicast
   ```

5. Filter the incoming route updates from the specified BGP neighbor.

```
device(config-bgp-ipv4u)# neighbor 10.2.3.4 prefix-list Routesfrom1234 in
```

This example specifies the **neighbor prefix-list** command, the IPv4 address of the neighbor, the prefix list named Routesfrom1234, and the keyword **in**.

6. Advertise ORF send capabilities.

```
device(config-bgp-ipv4u)# neighbor 10.2.3.4 capability orf prefixlist send
```

This example specifies the **neighbor capability orf prefixlist** command with the **send** keyword.

Example

The following example summarizes the commands in this procedure.

```
device# configure terminal
device(config)# ip prefix-list Routesfrom1234 deny 10.20.0.0/24
device(config)# ip prefix-list Routesfrom1234 permit 10.0.0.0/0 le 32
device(config)# 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 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 (except when a **continue** statement is used):

- 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.

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.

For more information about creating and using route maps, see the following commands in the *Extreme SLX-OS Command Reference*.

- continue
- match (route-map)
- route-map
- set as-path prepend

Configure BGP4 Route Map Matching on an AS-path

You can configure a route map that matches on a specified AS-path access control list (ACL).

Before You Begin
The AS-path ACL must be configured using the **ip as-path access-list** command.

Procedure

1. Access global configuration mode.

    ```
    device# configure terminal
    ```

2. Create a route map instance and allow a matching pattern.

    ```
    device(config)# route-map myaclroutemap1 permit 10
    ```

    This example creates a route map instance called myaclroutemap1 and allows a matching pattern of 10.

3. Configure the route map to match on an AS-path ACL.

    ```
    device(config-route-map-myaclroutemap1/permit/10)# match as-path myaspath
    ```

    This example configures the route map to match on the AS-path ACL called myaspath.

Example

The following example summarizes the commands in this procedure.

```
device# configure termnial
device(config)# route-map myaclroutemap1 permit 10
device(config-route-map-myaclroutemap1/permit/10)# match as-path myaspath
```

Configure BGP4 Route Matching on a Community ACL

You can configure a route map instance that matches on a specified community access control list (ACL).

Procedure

1. Access global configuration mode.

    ```
    device# configure terminal
    ```

2. Create a community ACL that permits traffic.

    ```
    device(config)# ip community-list standard 1 permit 123:2
    ```

    This example creates a standard community ACL named "1" that permits traffic for network 2 in autonomous system 123.

3. Create a route map instance and allow a matching pattern.

    ```
    device(config)# route-map mycommroutemap1 permit 10
    ```

    This example creates a route map instance called mycommroutemap1 and allows a matching pattern of 5.

4. Match the community ACL in the configured route-map instance.

    ```
    device(config-route-map-mycommroutemap1/permit/10)# match community 1
    ```

    This example matches the ACL named 1 in the route map instance.

Example

This example summarizes the commands in this procedure.

```
device# configure terminal
device(config)# ip community-list standard 1 permit 123:2
device(config)# route-map mycommroutemap1 permit 10
device(config-route-map-mycommroutemap1/permit/10)# match community 1
```

Configure BGP4 Route Map Matching on a Destination Network

You can configure a route map that matches on a destination network.

Procedure

1. Access global configuration mode.

   ```
   device# configure terminal
   ```

2. Create a route map instance and allow a matching pattern.

   ```
   device(config)# route-map mynetroutemap1 permit 10
   ```

   This example creates a route map instance named mynetroutemap1 and allows a matching pattern of 10.

3. Configure the route map to match on a specified prefix.

   ```
   device(config-route-map-mynetroutemap1/permit/10)# match ip address prefix-list mylist
   ```

   This example configures the route map to match on the prefix list titled mylist.

Example

This example summarizes the commands in this procedure.

```
device# configure terminal
device(config)# route-map mynetroutemap1 permit 10
device(config-route-map-mynetroutemap1/permit/10)# match ip address prefix-list mylist
```

Configure BGP4 Route Map Matching on a Static Network

You can configure route map that matches on a static network.

About This Task

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.

Procedure

1. Access global configuration mode.

   ```
   device# configure terminal
   ```

2. Create a route map instance and allow a matching pattern.

   ```
   device(config)# route-map mystaticroutemap3 permit 1
   ```

   This example creates a route map instance named mystaticroutemap3 and allows a matching pattern of 1.

3. Configure the route map to match on BGP4 static network routes.

   ```
   device(config-routemap-mystaticroutemap3/permit/1)# match protocol bgp static-network
   ```

4. Specify a BGP local-preference path attribute in the route map instance.

   ```
   device(config-routemap-mystaticroutemap3/permit/1)# set local-preference 150
   ```

5. Configure 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
   ```

6. Exit route map configuration mode.

   ```
   device(config-routemap-mystaticroutemap3/permit/1)# exit
   ```

7. Access BGP configuration mode.

```
device(config)# router bgp
```

8. Access IPv4 address family configuration mode.

```
device(config-bgp-router)# address-family ipv4 unicast
```

9. Filter the outgoing route updates to a BGP neighbor.

```
device (config-bgp-ipv4u)# neighbor 10.1.2.3 route-map out mystaticroutemap3
```

This example instructs the route map to filter outgoing route updates to 10.1.2.3 according to the attributes that are defined in mystaticroutemap3.

Example

The following example summarizes the commands in this procedure.

```
device# configure terminal
device(config)# 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)# router bgp
device(config-bgp)# address-family ipv4 unicast
device(config-bgp-ipv4u)# neighbor 10.1.2.3 route-map out mystaticroutemap3
```

Configure BGP4 Route Map Matching on a Next-hop Device

You can configure a route map that matches on a next-hop device.

Before You Begin
The prefix list must be configured using the **ip prefix-list** command.

Procedure

1. Access global configuration mode.

```
device# configure terminal
```

2. Create a route map instance and allow a matching pattern.

```
device(config)# route-map myhoproutemap1 permit 10
```

This example creates a route map instance named myhoproutemap1 and allows a matching pattern of 10.

3. Configure the route map to match IP next-hop match conditions for a prefix list.

```
device(config-route-map-myhoproutemap1/permit/10)# match ip next-hop prefix-list mylist
```

This example configures the route map to match next-hop match conditions for the prefix list titled mylist.

Example

The following example summarizes the commands in this procedure.

```
device# configure terminal
device(config)# route-map myhoproutemap1 permit 10
device(config-route-map-myhoproutemap1/permit/10)# match ip next-hop prefix-list mylist
```

Configure BGP4 Route Map Matching on an Interface

You can configure a route map that matches on an Ethernet interface, a loopback interface, or a VE interface.

Procedure

1. Access global configuration mode.

   ```
   device# configure terminal
   ```

2. Create a route map instance and allow a matching pattern.

   ```
   device(config)# route-map myintroutemap1 permit 99
   ```

   This example creates a route map instance named myintroutemap1 and allows a matching pattern of 99.

3. Configure the route map to match on an interface.

   ```
   device(config-route-map-myintroutemap1/permit/99)# match interface ethernet 1/1
   loopback 1
   ```

   This example configures the route map instance to match on Ethernet 1/1 and loopback 1.

Example

The following example summarizes the commands in this procedure.

```
device# configure terminal
device(config)# route-map myintroutemap1 permit 99
device(config-route-map-myintroutemap1/permit/99)# match interface ethernet 1/1 loopback 1
```

Setting a BGP4 route MED to equal the next-hop route IGP metric

You can set the MED of the advertised BGP route to the IGP metric of the next hop route. This setting works only with the outbound route map used with a BGP neighbor. You can use this configuration with BGP shortcuts (IPoMPLS) and IPv4 or IPv6 address families in default or user VRFs.

About This Task

The following task shows how to set the route MED to the same value as the IGP metric of the BGP4 next-hop route.

Procedure

1. From privileged EXEC mode, enter global configuration mode.

   ```
   device# configure terminal
   ```

2. Configure a route map to permit the destination network.

   ```
   device(config)# route-map medroute permit 1
   ```

   This example creates a route map named medroute that allows a matching pattern and enters configuration mode for the route-map sequence number 1.

   > **Note**
   > Route-map **deny** sequences are not advertised; **deny** configurations are ignored.

3. Set the metric type for the routes to the same value as the IGP metric of the BGP4 next-hop route.

   ```
   device(config-routemap-medroute/permit/1)# set metric-type internal
   ```

4. Return to privileged EXEC mode.

```
device(config-routemap-medroute/permit/1)# end
```

5. From privileged EXEC mode, enter global configuration mode.

```
device# configure terminal
```

6. Access BGP router configuration mode.

```
device(config)# router bgp
```

7. Specify the BGP autonomous system number (ASN) where the device resides.

```
device(config-bgp-router)# local-as 666
```

8. Specify the autonomous system (AS) in which a remote neighbor resides

```
device(config-bgp-router)# neighbor 10.11.12.13 remote-as 100
```

9. Enable BGP IPv4 address-family configuration mode.

```
device(config-bgp-router)# address-family ipv4 unicast
```

10. Apply the medroute route map to an outgoing route from 10.11.12.13.

```
device(config-bgp-ipv4u)# neighbor 10.11.12.13 route-map out medroute
```

11. Return to privileged EXEC mode.

```
device(config-bgp-ipv4u)# end
```

Example

The following example shows the configuration in the previous steps.

```
device# configure terminal
device(config)# route-map medroute permit 1
device(config-routemap-medroute/permit/1)# set metric-type internal
device(config-routemap-medroute/permit/1)# end
device# configure terminal
device(config)# router bgp
device(config-bgp-router)# local-as 666
device(config-bgp-router)# neighbor 10.11.12.13 remote-as 100
device(config-bgp-router)# address-family ipv4 unicast
device(config-bgp-ipv4u)# neighbor 10.11.12.13 route-map out medroute
```

What to Do Next
To display the MED for the route, use the **show ip bgp neighbors advertised-routes** command.

```
device# show ip bgp neighbors 10.11.12.13 advertised-routes
       There are 1 routes advertised to neighbor 10.11.12.13
Status A:AGGREGATE B:BEST b:NOT-INSTALLED-BEST E:EBGP I:IBGP L:LOCAL x:BEST-
EXTERNAL
       Prefix          Next Hop               MED        LocPrf     Weight Status
1      99.99.99.0/24   10.11.12.13            14                    0      BEx
         AS_PATH: 666 100
```

If any dynamic change in the IGP cost of next hop occurs after applying the outbound route map to the BGP neighbor, the change is not reflected in the MED field implicitly. Use **clear ip bgp neighbor** command with the **soft-outbound** option to reflect the change.

```
device# clear ip bgp neighbor 10.11.12.13 soft-outbound
```

Route Map Continue Statement for BGP4 Routes

A continue statement in a route map directs program flow to skip route map instances to another, user-specified instance. When 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 run the set clauses of the matched instances.

The system does not update routes if it finds no matches in the route map instances or if it encounters a deny condition. When a matched instance contains a deny statement, the current traversal is terminated. None of the updates in the set statements of the matched instances in current and previous traversals are applied to the routes.

The continue statement feature supports a more programmable route map configuration and route filtering scheme for BGP4 peering. The feature 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 in the same route map.

This feature applies only to BGP4 routes. Continue statements are ignored for protocols other than BGP4.

Use 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.

Procedure

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

   ```
   device# configure terminal
   ```

2. 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)# route-map mcontroutemap1 permit 1
   ```

3. 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
   ```

4. 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
   ```

5. 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
   ```

6. Enter the **exit** command to exit route map configuration mode.

   ```
   device(config-routemap-mycontroutemap/permit/1)# exit
   ```

7. 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)# route-map mcontroutemap1 permit 2
```

8. 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
```

9. 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
```

10. 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
```

Example

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)# route-map mcontroutemap1 permit 1
device(config-routemap-mycontroutemap1/permit/1)# match metric 10
device(config-routemap-mycontroutemap1/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)# route-map mcontroutemap1 permit 2
device(config-routemap-mycontroutemap1/permit/2)# match tag 10
device(config-routemap-mycontroutemap1/permit/2)# set weight 20
```

BGP4 Confederations

A large autonomous system (AS) can be divided into multiple sub-autonomous (sub-AS) systems and grouped into one BGP4 confederation.

Each sub-AS system must be uniquely identified in the confederation AS by a sub-AS system number. In each sub-AS system, all the rules of internal BGP (iBGP) apply. For example, all BGP routers in the sub-AS system must be fully meshed. Although eBGP is used between sub-AS systems, the sub-AS systems in the confederation exchange routing information like iBGP peers. Next hop, Multi Exit Discriminator (MED), and local preference information is preserved when it crosses sub-AS system boundaries. To the outside world, a confederation looks like one AS.

The AS path list is a loop-avoidance mechanism used to detect routing updates that leave one sub-AS system and attempt to re-enter the same sub-AS system. A routing update that attempts to re-enter the sub-AS system from which it originated is detected because the sub-AS system sees its own system number listed in the update's AS path.

Figure 13: Example BGP4 confederation

In this example, four devices are configured into two sub-AS systems, each containing two of the devices. The sub-AS are members of confederation 10. Devices in a sub-AS must be fully meshed and communicate using iBGP. In this example, devices A and B use iBGP to communicate. Devices C and D also use IBGP. However, the sub-AS systems communicate with one another using eBGP. For example, device A communicates with device C using eBGP. The devices in the confederation communicate with other autonomous systems using eBGP.

Devices in other ASs are unaware that devices A through D are configured in a confederation. In fact, when devices in confederation 10 send traffic to devices in other ASs, the confederation ID is the same as the AS number for the devices in the confederation. Thus, devices in other ASs see traffic as coming from AS 10 and are unaware that the devices in AS 10 are subdivided into sub-AS systems in a confederation.

Configure a BGP4 Confederation

A BGP4 confederation consists of multiple sub-autonomous systems.

Procedure

1. Access global configuration mode.

   ```
   device# configure terminal
   ```

2. Access BGP configuration mode.

   ```
   device(config)# router bgp
   ```

3. Identify the sub-autonomous system (sub-AS) in which your device resides.

```
device(config-bgp-router)# local-as 65520
```

4. Assign an ASN as a BGP confederation identifier.

```
device(config-bgp-router)# confederation identifier 100
```

Sub-AS 65520 is now part of a confederation that belongs to autonomous system 100.

5. Specify the ASNs of all BGP peers that you want to add to the confederation.

```
device(config-bgp-router)# confederation peers 65520 65521 65522
```

Example

The following example summarizes the commands in this procedure.

```
device# configure terminal
device(config)# 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.

Define a BGP4 Extended Community Filter

You must define a BGP4 extended community filter before you can apply the filter.

Procedure

1. Access global configuration mode.

```
device# configure terminal
```

2. Specify an extended community list instance.

```
device(config)# ip extcommunity-list 1 permit soo 123:2
```

This example specifies the extended community list and permits site of origin (SOO) from network 2 in autonomous system 123.

> **Note**
> The **ip-extcommunity-list** command supports a range of extended instances from 100 through 500, beyond the standard range of 1 through 99.

3. Create a route map instance and enter route-map configuration mode.

```
device(config)# route-map extComRmap permit 10
```

This example creates the extComRmap route map that permits (allows) a matching pattern, with an instance ID of 10.

4. Configure the route map to match the extended community list number.

```
device(config-route-map-extComRmap/permit/10)# match extcommunity 1
```

5. Exit route-map configuration mode.

```
device(config-route-map-extComRmap/permit/10)# exit
```

6. Define the route map and enter route-map configuration mode.

```
device(config)# route-map sendExtComRmap permit 10
```

7. Specify the route target (RT) extended community attribute.

```
device(config-route-map-sendExtComRmap/permit/10)# set extcommunity rt 3:3
```

This example specifies the **set extcommunity** command, the **rt** keyword, and the *extcommunity value* variable.

8. Set the extended community attribute in the route-map instance.

```
device(config-route-map-sendExtComRmap/permit/10)# set extcommunity soo 2:2
```

This example specifies the SOO extended community attribute with the **soo** keyword and the *extcommunity value* variable.

Example

The following example summarizes the commands in the procedure.

```
device# configure terminal
device(config)# ip extcommunity-list 1 permit soo 123:2
device(config)# 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)# 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
```

Apply a BGP4 Extended Community Filter

You can apply an extended community filter to a neighbor through a route map.

Before You Begin
Define a BGP4 Extended Community Filter on page 161.

Procedure

1. Access global configuration mode.

   ```
   device# configure terminal
   ```

2. Define the matching conditions for the extended community.

   ```
   device(config)# ip extcommunity-list 1 permit rt 123:2
   ```

   This example specifies that community list 1 permits routes from network 2 in autonomous system 123.

3. Access BGP configuration mode.

   ```
   device(config)# router bgp
   ```

4. Specify the autonomous system in which your device resides.

   ```
   device(config-bgp-router)# local-as 1000
   ```

5. Specify the autonomous system in which the remote neighbor resides.

   ```
   device(config-bgp-router)# neighbor 10.1.2.3 remote-as 1001
   ```

6. Access IPv4 address family configuration mode.

   ```
   device(config-bgp-router)# address-family ipv4 unicast
   ```

7. Enable the exchange of information with the neighbor.

   ```
   device(config-bgp-ipv4u)# neighbor 10.1.2.3 activate
   ```

8. Apply a route map to incoming routes.

   ```
   device(config-bgp-ipv4u)# neighbor 10.1.2.3 route-map in extComRmapt
   ```

   This example uses the **in** keyword to apply route map extComRmapt.

9. Apply a route map to outgoing routes.

   ```
   device(config-bgp-ipv4u)# neighbor 10.1.2.3 route-map out sendExtComRmap
   ```

   This example uses the **out** keyword to apply route map sendExtComRmap.

10. 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
    ```

    This example uses the **send-community both** keywords.

Example

The following example summarizes the commands in this procedure.

```
device# configure terminal
device(config)# ip extcommunity-list 1 permit rt 123:2
device(config)# 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 10.1.2.3 send-community both
```

BGP Large Communities

The BGP Large Communities attribute (RFC 8092) supports an optional transitive path attribute of variable length, overcoming previous length limitations. All routes with this attribute belong to the communities specified in the attribute.

Introduction

BGP (RFC 4271) implementations typically support a routing policy language to control the distribution of routing information. Network operators attach BGP communities to routes to associate particular properties with these routes.

A BGP Communities attribute (RFC 1997) is a variable-length attribute consisting of a set of one or more four-octet values, each of which specifies a community. A common use of the individual values of this attribute splits this 32-bit value into two 16-bit values. The most significant word is interpreted as an Autonomous System Number (ASN), and the least significant word is a locally defined value whose meaning is assigned by the operator of the AS in the most significant word.

Since the adoption of four-octet ASNs, the BGP Communities attribute cannot accommodate the encoding, because a two-octet word cannot fit a four-octet ASN.

To address this shortcoming, a BGP Large Communities attribute is encoded as an unordered set of one or more 12-octet values, each consisting of a four-octet Global Administrator field and two 4-octet operator-defined fields. Each operator-defined file can be used to denote properties or actions significant to the operator of the AS assigning the values.

BGP Large Communities attribute details

Each BGP Large Community value is encoded as a 12-octet quantity, as follows:

```
0 1 2 3 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                         Global Administrator                  |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                          Local Data Part 1                    |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

```
|                              Local Data Part 2                           |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Field	Description
Global Administrator	A four-octet namespace identifier. Allows different ASes to define BGP Large Communities without collision. This field SHOULD be an ASN, in which case the Local Data Parts are to be interpreted as defined by the owner of the ASN.
Local Data Part 1	A four-octet operator-defined value. Represents a function identifier.
Local Data Part 2	A four-octet operator-defined value. Represents a parameter value.

Important

The use of Reserved ASNs (0 [RFC 7607], 65535, and 4294967295 [RFC 7300] is **not** a best practice.

There is no significance to the order in which twelve-octet Large Community Attribute values are encoded in a Large Communities attribute. A BGP speaker can transmit them in any order.

In canonical notation, each Large Community value can be summarized as "ASN:Function:Parameter". The function of a community can be divided into two categories for the treatment of routes:

- **Informational Communities**: Labels for attributes such as the origin of the route announcement, the nature of the relation with an External BGP (eBGP) neighbor, or the intended propagation audience.

- **Action Communities**: Added as labels to request that a route be treated in a particular way in an AS

For more information on the use of BGP Large Communities, see RFC 8195.

BGP Large Community configuration examples

This section presents a variety of configuration examples.

Defining a large community standard ACL

The following example creates a standard large-community list.

```
device# configure terminal
device(config)# ip large-community-list standard my_std_list seq 10 permit 64497:1:528
```

Defining a large community extended ACL

The following example creates an extended community list with a regular expression.

```
device# configure terminal
device(config)# ip large-community-list extended my_ext_list seq 10 permit 64497:*:*
```

Defining a large community list in a route map

Each route-map instance can have only one set large-community directive in which you can define a maximum of 32 large communities. The following example sets a large-community list in route-map instances to advertise routes with large-community attributes.

```
device# configure terminal
device# configure terminal
device(config)# route-map myroutes permit 10
device(config-route-map-myroutes/permit/10)# set large-community 64497:1:528
device(config-route-map-myroutes/permit/10)# exit
device(config)# route-map myroutes permit 20
device(config-route-map-myroutes/permit/20)# set large-community 64497:5:528
device(config-route-map-myroutes/permit/20)# exit
device(config)# route-map myroutes permit 30
device(config-route-map-myroutes/permit/30)# set large-community 64497:6:528
```

The following example configures multiple large communities for a set large-community directive.

```
device# configure terminal
device(config)# route-map myroutes permit 10
device(config-route-map-myroutes/permit/10)# set large-community 64497:1:528 64497:2:529
```

The following example sets a large-community list in a route-map instance and appends updates to existing attributes.

```
device# configure terminal
device(config)# route-map myroutes permit 10
device(config-route-map-myroutes/permit/10)# set large-community 64497:1:528 additive
```

Defining a route map matching a large-community access list

The following example shows how to configure matching based on a large-community access list named ABCPath for a route map named myroutes.

```
device# config terminal
device(config)# route-map myroutes permit 10
device(config-route-map myroutes/permit/10)# match large-community ABCPath
```

The following example shows how to configure matching based on a large-community access list named lcstdacl1 with an exact match for a route map named myroutes.

```
device# config terminal
device(config)# route-map myroutes permit 10
device(config-route-map myroutes/permit/10)# match large-community lcstdacl1 exact-match
```

Defining a route map to delete large community attributes

The following example deletes community attributes specified in the ACL from the route update. instance.

```
device# configure terminal
device(config)# route-map myroutes permit 10
device(config-route-map-myroutes/permit/10)# set large-community-list
mylargecommunitylist delete
```

Enabling the sending of large community attributes in BGP attributes

> **Note**
>
> Configure the BGP neighbor before executing the **neighbor send community** command. If you use the **neighbor send-community** command after a BGP session has been established, the BGP session must be reset for this change to take effect.

The following example sends the large community attributes in BGP attributes.

```
device# configure terminal
device(config)# router bgp
device(config-bgp)# address-family ipv4 unicast
(config-bgp-ipv4u)# neighbor 10.11.12.13 send-community large
```

The following example sends all communities.

```
device# configure terminal
device(config)# router bgp
device(config-bgp)# address-family ipv4 unicast
(config-bgp-ipv4u)# neighbor 10.11.12.13 send-community all
```

The following example adds the configured route-map for a peer route-map out.

```
device# configure terminal
device(config)# router bgp
device(config-bgp)# address-family ipv4 unicast
(config-bgp-ipv4u)# neighbor 10.11.12.13 route-map out myroutes
```

BGP Flowspec

BGP flow specification (flowspec) enables a device to instruct BGP neighbors to apply a specific traffic policy or flowspec rule. Neighbors, at their discretion, can apply the policy to take a particular action on traffic or to filter, for example, distributed denial of service (DDoS) attacks.

> **Note**
>
> Only non-VPN IPv4 BGP flowspec is supported (in both default and non-default VRFs).

A traffic policy that an IP router uses to forward traffic consists of match criteria and corresponding actions. Match criteria operate on different fields in a packet such as source or destination IP prefixes, IP protocol, and transport-layer port numbers. The corresponding actions can be tasks such as rate limit, filter, and redirect.

BGP flowspec is an n-tuple that consists of a number of matching criteria that define an aggregate IP traffic flow specification. The associated set of actions are advertised by using BGP.

The Extreme Networks implementation of BGP flowspec supports RFC 5575 and conforms to the following standards:

- draft-ietf-idr-bgp-flowspec-oid-06: Revised Validation Procedure for BGP Flow Specifications
- draft-ietf-idr-flowspec-redirect-ip-02: BGP Flow-Spec Redirect to IP Action

BGP flowspec can be applied to DDoS attack filtering. Traditional methods of filtering DDoS attacks include advertising a black hole route to the destination under attack and advertising the destination under attack with a special community that sets the nexthop to a discard nexthop. The disadvantages of traditional approaches include a lack of granularity and the mixing of routing and filtering

information. BGP flowspec facilitates a more granular approach to DDoS attack filtering, allowing you to propagate policies that filter and police DDoS attacks in a service-provider environment where mitigation points may be in different autonomous systems.

The following figure describes DDoS attack filtering using BGP flowspec.
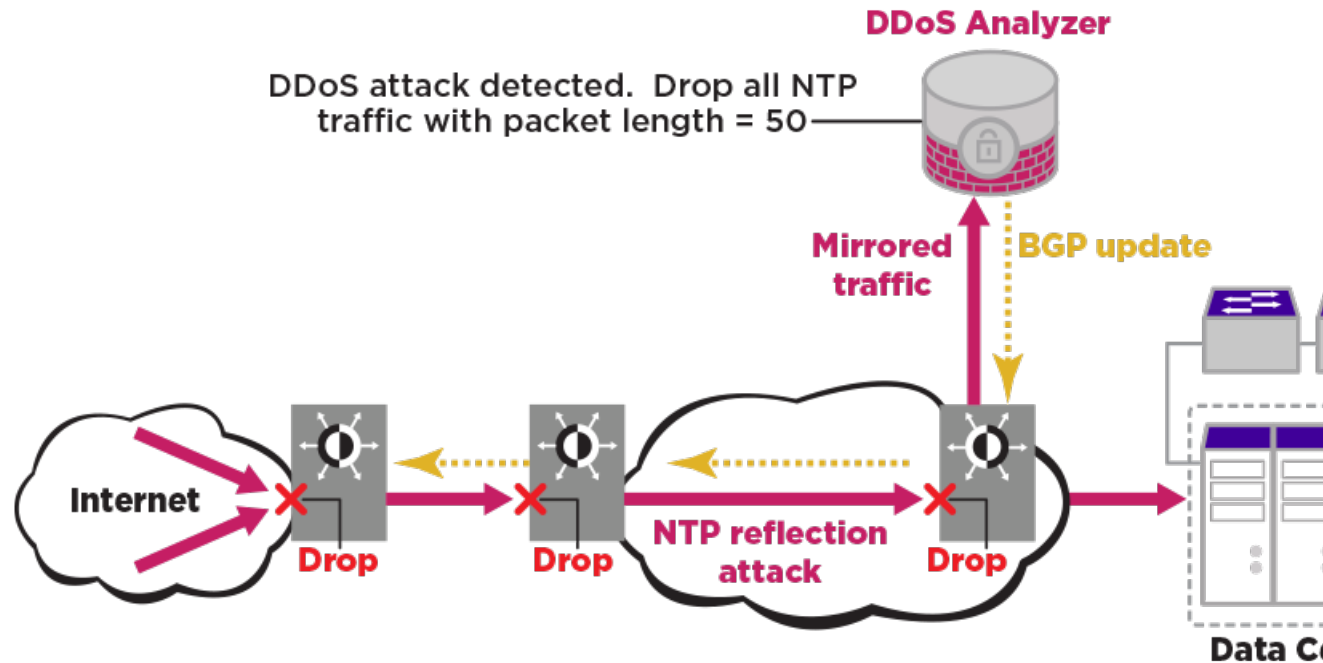
In the figure, the DDoS Analyzer identifies the threat and generates the filter and corresponding actions according to the flowspec rules. The DDoS appliance then sends out the filter and policing function information as BGP network layer reachability information (NLRI).

Alternatively, you can use the CLI to configure the policy in one of the BGP speakers, such as a route reflector, for propagation to other BGP peer devices.

Figure 14: DDoS attack filtering

Distribution of Flowspec Rules by BGP

BGP distributes flow specification (flowspec) rules by using a flow specification Network Layer Reachability Information (NRLI) type.

A BGP application is identified by a specific AFI-SAFI pair. Non-VPN IPv4 BGP flowspec has the following AFI-SAFI pair:

* AFI (Address Family Identifier) = 1 (IPv4)
* SAFI (Subsequent Address Family Identifier) = 133

The flow specification NRLI type consists of several optional sub-component types. These sub-component types form the n-tuple of the matching criteria. A specific packet is considered to match the

flow specification when it matches the components types in the specification. You can define the following sub-component types or tuples.

Table 20: BGP flowspec NLRI sub-component types

BGP flowspec NLRI type	Description	Encoding
Type 1	Destination Prefix	<type (1 octet), prefix length (1 octet), prefix>
Type 2	Source Prefix	<type (1 octet), prefix-length (1 octet), prefix>
Type 3	IP Protocol (IPv4)	<type (1 octet), [op, value]+>
Type 4	Port	<type (1 octet), [op, value]+>
Type 5	Destination port	<type (1 octet), [op, value]+>
Type 6	Source port	<type (1 octet), [op, value]+>
Type 7	ICMP type	<type (1 octet), [op, value]+>
Type 8	ICMP code	<type (1 octet), [op, value]+>
Type 9	TCP flags (CWR, ECE, URG, ACK, PSH, RST, SYN, FIN)	<type (1 octet), [op, bitmask]+>
Type 10	Packet length	<type (1 octet), [op, value]+>
Type 11	DSCP	<type (1 octet), [op, value]+>
Type 12	Fragment (LF, FF, IsF, DF)	<type (1 octet), [op, bitmask]+>

BGP Flowspec Traffic Filtering Actions

The default action for filtering BGP flow specification (flowspec) traffic is to accept the IP traffic that matches the specification.

The following extended community values specify particular actions for traffic matching the flow specification in the NLRI.

Table 21: BGP flowspec traffic filtering actions

Type	Extended community value	Encoding
0x8006	traffic-rate	2-byte as#, 4-byte float
0x8007	traffic-action	bitmask
0x8009	traffic-marking	DSCP value
0x0800 or 0x080C	redirect IPv4	4 byte IPv4

BGP Flowspec Considerations

When a device is downgraded to a release that is earlier than 18r.2.00, the BGP flowspec feature does not work.

Data plane considerations

- Only non-VPN IPv4 BGP flowspec is supported.

- Match types other than the 12 BGP flowspec NLRI sub-component types described in Distribution of Flowspec Rules by BGP on page 168 are considered unknown. A flowspec NLRI that contains an unknown match type is considered invalid and is not advertised or installed in the hardware.

- The following TCP flags are not supported:
  - Explicit Congestion Notification Echo (ECE)
  - Congestion Window Reduced (CWR)

- Two-byte TCP flags are not supported.

- When a TCP flag sub-component is larger than one byte, a RASlog message is triggered and it is not installed in the hardware. However, it is advertised to peer devices.

- Only the IsF bit is supported for BGP flowspec NLRI sub-component type 12 (Fragment). DF, FF, and LF bit functionality is not supported.

- When the match criteria of a stanza in a flowspec route map requires an NLRI length that is greater than 4095, the route map is not installed or advertised by BGP.

- Actions other than the four BGP flowspec traffic filtering actions described in BGP Flowspec Traffic Filtering Actions on page 169 are considered unknown. For a flowspec NLRI that contains an unknown action:
  - The unknown action (user-defined extended community or unknown action) is not installed.
  - The remaining flowspec rules are installed.
  - The flowspec NLRI is advertised to peers with the unknown extended communities.

- Copy or mirror action is not supported.

- When a rate-limiting action is set under a BGP flowspec rule, the operational rate value may differ from the rate value specified in the flowspec rule because operational values are selected in multiples of 22 kbits per second.

> **Note**
> When the rate-limiting action under a BGP flowspec rule is set to a value that is lower than 22 kbits per second, matched data traffic is dropped.

- With the default TCAM profile, BGP flowspec routes configured with the following match criteria can be advertised but not installed in the hardware:
  - IP fragment
  - Packet length
  - ICMP code
  - ICMP type

  To maximize the BGP flowspec match criteria and actions supported in the hardware, a BGP flowspec profile must first be enabled in the hardware by the `profile tcam border-routing` command.

- Traffic-marking (set dscp) is not supported in the default TCAM profile.

- With the default TCAM profile, flowspec can be used only when there are no user-defined VRFs.

- BGP flowspec rules are applied on all Layer 3 interfaces of the specified VRF.
- IPv4 BGP flowspec rules are applied only to IPv4 data traffic. They are not applied to IPv6 data traffic.
- CAM sharing is not supported in the border routing TCAM profile.
- Several match commands, such as **`match dscp`**, support a **`range`** option. Use the **`range`** option with caution because many TCAM entries may be created when the rules are expanded.
- Redirection to multiple nexthop addresses is not supported. When multiple redirect nexthops are configured, only the first valid, reachable nexthop is used. If the first nexthop becomes invalid or unreachable, then the next configured, valid, reachable nexthop is used.
- Matching of traffic flow with subsequent flowspec rules (terminal-action) is not supported.
- Each flowspec rule may be expanded to several access control list (ACL) rules in TCAM. When TCAM is full, a RASlog message is displayed. However, the BGP transit functionality of advertising and receiving flowspec rules can continue when TCAM is full or a flowspec rule is not installed in TCAM.
- BGP flowspec rules are prioritized over policy-based routing rules. Policy-based routing rules are prioritized over ACL rules.

Control plane considerations

- A maximum of 1,000 (configured and received) flowspec rules is a best practice.

> **Note**
> In BGP RIB-in, there is no hard-coded limit for BGP flowspec routes. When sufficient memory exists, BGP RIB-in receives more routes. However, a maximum of 1,000 routes is a best practice.

- Any match criterion or traffic action in a BGP flowspec route that is not supported in the hardware is still received and advertised by BGP.
- Dampening and soft reconfiguration is not supported for the BGP flowspec address family.
- The nexthop path attribute is not added to BGP flowspec routes by default.

Workflow for Configuring BGP Flowspec

Configuration of BGP flowspec includes optionally configuring the border-routing profile in TCAM, enabling the BGP flowspec address family and address-family neighbors, and generating and distributing the flowspec rules.

Configuration of BGP flowspec consists of the following high-level tasks.

1. (Optional) Enable the border-routing profile in TCAM. The default TCAM profile does not support the following flowspec match criteria.
   - Fragment
   - ICMP code
   - ICMP type
   - Packet length

   Therefore, for maximum support of BGP flowspec match criteria and actions, enable the border-routing profile. For more information, see Enable the Border-Routing TCAM Profile on page 173.
2. (Optional) Collect forward or drop statistics.

When statistics are enabled (with the `ip flowspec rules statistics` command), statistics are collected only for traffic that matches BGP flowspec rules.

To collect forward or drop statistics, use the `profile counters` command, specifying either the `counter-profile-1` or the `counter-profile-4` option. For more information, see Enable the Border-Routing TCAM Profile on page 173.

3. Enable the BGP flowspec address family and activate the neighbors that are needed to advertise or receive the flowspec rules. For more information, see Enable the BGP Flowspec Address Family and Activate Neighbors on page 175.

4. Configure the BGP flowspec rules. Rules are configured as different sequences of a route map. For more information, see Configure BGP Flowspec Rules on page 173.

5. (Optional) When BGP flowspec validation is not needed, disable it at the neighbor or peer-group level with the `neighbor flowspec validation` command, or at the address-family level with the `flowspec validation` command.

6. (Optional) When a peer device from another vendor supports a different implementation of the redirect nexthop action, use the CLI to configure the respective interoperational parameters for generating local rules based on the compatibility of the peer device.

7. Pass the route map to BGP by using the `distribute` command in BGP address-family IPv4 flowspec configuration mode. For more information, see Distribute BGP Flowspec Rules on page 176.

Duplicate Stanzas in a BGP Flowspec Route Map

Duplicate stanzas have the same match statements. BGP installs only one local flowspec route for a specific key (match statement) and has rules for handling duplicate stanzas.

- BGP traverses the flowspec route map in ascending order of sequence number when:
  ◦ The route map is first applied with the `distribute` command
  ◦ Modifications are made to an active route map.
- While traversing each stanza, BGP attempts to prepare or modify a local flowspec route.
- When the `distribute` command is used to activate a route map that contains duplicate stanzas, the stanza with the higher sequence number becomes active only if it has different `set` commands. Otherwise the stanza with the lower sequence number remains active. For example: Stanzas A and B are duplicates. Stanza A has a lower sequence number than stanza B. Therefore:
  ◦ When A and B have same `match` and `set` command content, A is installed, B is not allowed to be active, and a RASlog message is triggered.
  ◦ When A and B have same `match` command content but different `set` command content, A first becomes active, then B replaces A, and a RASlog message is triggered.
- When a change is made to the `match` content of an active stanza, the stanza with the lowest sequence number from the remaining duplicates is selected as active. For example: a route map has three duplicate stanzas (A, B, and C). For these stanzas, the `match` command content is the same but the `set` command content differs. Also, the sequence number of A is lower than the sequence number of B, which is lower than the sequence number of C. Therefore, assuming that stanza C is active:
  ◦ When the `match` content in stanza C is modified, the stanza with the lowest sequence number among stanza C's former duplicates (A) is selected as active along with the new stanza C and a RASlog message is triggered.

◦   When stanza C is removed, the stanza with the lowest sequence number among stanza C's former duplicates (A) is selected as active and a RASlog message is triggered.

Enable the Border-Routing TCAM Profile

Enabling the border-routing Ternary Content-Addressable Memory (TCAM) profile ensures that the maximum number of BGP flowspec match criteria and actions is supported in the hardware.

About This Task

➡ Important
   This procedure requires a system reload to activate the profile configuration.

Procedure

1. From privileged EXEC mode, enter global configuration mode.

```
device# configure terminal
```

2. Enter hardware configuration mode.

```
device(config)# hardware
```

3. Enable a BGP flowspec profile in TCAM.

```
device(config-hardware)# profile tcam border-routing
```

4. (Optional) Enable forward or drop statistics using one of the following options.

```
device(config-hardware)# profile counters counter-profile-1
device(config-hardware)# profile counters counter-profile-4
```

5. Access privileged EXEC mode.

```
device(config-hardware)# exit
```

6. Activate the profile configuration.

```
device# reload system
```

Example

The following example summarizes the commands in this procedure.

```
device# configure terminal
device(config)# hardware
device(config-hardware)# profile tcam border-routing
device(config-hardware)# profile counters counter-profile-1
device(config-hardware)# exit
device# reload system
```

Configure BGP Flowspec Rules

BGP flowspec rules consist of match criteria and traffic actions that are configured under a route-map sequence number. Each sequence number or stanza in the route map becomes a flowspec rule.

About This Task

The following task shows how to configure a selection of BGP flowspec rules. Your configuration may involve different combinations of match criteria and traffic filtering actions.

Procedure

1. From privileged EXEC mode, enter global configuration mode.

   ```
   device# configure terminal
   ```

2. Create a route map for BGP flowspec rules that allows a matching pattern.

   ```
   device(config)# route-map flowspec_map permit 56
   ```

   This example creates a route map named flowspec_map that allows a matching pattern and enters configuration mode for the route-map sequence number 56.

   > **Note**
   > Route-map **deny** sequences are not advertised as flowspec rules and **deny** configurations are ignored.

3. Configure matching based on destination prefix.

   ```
   device(config-route-map-flowspec_map/permit/56)# match ip destination-address
   10.2.3.0/24
   ```

4. Configure matching based on protocol number.

   ```
   device(config-route-map-flowspec_map/permit/56)# match protocol eq 30
   ```

5. Configure matching based on a range of protocol numbers.

   ```
   device(config-route-map-flowspec_map/permit/56)# match protocol range 40 50
   ```

6. Configure matching based on traffic having all specified TCP flags.

   ```
   device(config-route-map-flowspec_map/permit/56)# match tcp-flags all not-fin ack
   ```

   In this example, because the **all** option is specified, a match occurs when traffic matches all of the specified options (**not-fin** and **ack**).

7. Configure matching based on traffic having any of the specified TCP flags.

   ```
   device(config-route-map-flowspec_map/permit/56)# match tcp-flags any cwr urg
   ```

   Because the **any** option is specified in this example, a match occurs when traffic matches any other specified option: **cwr** or **urg**.

8. Configure matching based on traffic having an ICMP type that is not equal to 5.

   ```
   device(config-route-map-flowspec_map/permit/56)# match ip icmp-type neq 5
   ```

9. Configure a committed information rate (CIR) action.

   ```
   device(config-route-map-flowspec_map/permit/56)# set police cir 0
   ```

10. Configure a mirroring destination.

    ```
    device(config-route-map-flowspec_map/permit/56)# set ip mirror 10.67.67.9
    ```

11. Return to privileged EXEC mode.

    ```
    device(config-route-map-flowspec_map/permit/56)# end
    ```

12. If you configured a route map for the first time, activate the BGP flowspec rules. For more information, see

Example

The following example summarizes the commands in this task.

```
device# configure terminal
device(config)# route-map flowspec_map permit 56
device(config-route-map-flowspec_map/permit/56)# match ip destination-address 10.2.3.0/24
```

```
device(config-route-map-flowspec_map/permit/56)# match protocol eq 30
device(config-route-map-flowspec_map/permit/56)# match protocol range 40 50
device(config-route-map-flowspec_map/permit/56)# match tcp-flags all not-fin ack
device(config-route-map-flowspec_map/permit/56)# match tcp-flags any cwr urg
device(config-route-map-flowspec_map/permit/56)# match ip icmp-type neq 5
device(config-route-map-flowspec_map/permit/56)# set police cir 0
device(config-route-map-flowspec_map/permit/56)# set ip mirror 10.67.67.9
```

Enable the BGP Flowspec Address Family and Activate Neighbors

Enabling the BGP flowspec IPv4 address family and activating neighbors under this address family
supports propagation of a BGP flowspec policy.

Procedure

1. From privileged EXEC mode, enter global configuration mode.

   ```
   device# configure terminal
   ```

2. Enable BGP routing and enter BGP configuration mode.

   ```
   device(config)# router bgp
   ```

3. Specify the autonomous system number (ASN) in which the device resides.

   ```
   device(config-bgp-router)# local-as 666
   ```

4. Specify the ASN in which a remote neighbor resides.

   ```
   device(config-bgp-router)# neighbor 10.61.61.6 remote-as 666
   ```

5. Enable the IPv4 unicast address family in the VRF.

   ```
   device(config-bgp-router)# address-family ipv4 unicast vrf red
   ```

   This example shows how to enable the IPv4 unicast address family in a VRF instance named red.

   > **Note**
   > The IPv4 unicast address family is always configured on the default VRF.

6. Enable the flowspec IPv4 address family.

   ```
   device(config-bgp-router)# address-family ipv4 flowspec vrf red
   ```

   This example shows how to enable IPv4 flowspec address family in a VRF instance named red.

   > **Note**
   > When a VRF instance is not specified, IPv4 flowspec address family is enabled in the
   > default VRF.

7. Activate the flowspec route exchange with the remote neighbor under the BGP flowspec IPv4
   address family.

   ```
   device(config-bgp-ipv4fs)# neighbor 10.61.61.6 activate
   ```

8. (Optional) When the receiving device supports the **0x0800** extended-community type or the **nlri**
   holder (place where the nexthop IP address is encoded in the BGP update packet) or both, configure
   the flowspec redirect holder or nexthop action type for the neighbor.

   ```
   device(config-bgp-ipv4fs)# neighbor 10.61.61.6 flowspec redirect holder nrli next-hop
   type 0x0800
   ```

9. After you are done activating neighbors, configure the distribution of BGP flowspec rules. For more
   information, see Distribute BGP Flowspec Rules on page 176.

Example

The following example summarizes the commands in this task.

```
device# configure terminal
device(config)# router bgp
device(config-bgp-router)# local-as 666
device(config-bgp-router)# neighbor 10.61.61.6 remote-as 666
device(config-bgp-router)# address-family ipv4 unicast vrf red
device(config-bgp-router)# address-family ipv4 flowspec vrf red
device(config-bgp-ipv4fs)# neighbor 10.61.61.6 activate
device(config-bgp-ipv4fs)# neighbor 10.61.61.6 flowspec redirect holder nlri next-hop
type 0x0800
```

Distribute BGP Flowspec Rules

BGP distributes flowspec rules when a route map that is configured with flowspec rules is passed to BGP for advertisement by BGP neighbors.

Before You Begin

Before configuring distribution, complete the following tasks:

- Configure BGP Flowspec Rules on page 173
- Enable the BGP Flowspec Address Family and Activate Neighbors on page 175

Procedure

1. From privileged EXEC mode, enter global configuration mode.

   ```
   device# configure terminal
   ```

2. Enable BGP and enter BGP configuration mode.

   ```
   device(config)# router bgp
   ```

3. Enter BGP address-family IPv4 flowspec configuration mode.

   ```
   device(config-bgp-router)# address-family ipv4 flowspec
   ```

4. Distribute a route map that is already configured with the BGP flowspec rules.

   ```
   device(config-bgp-ipv4fs)# distribute flowspec-map
   ```

   This example shows how to distribute a route map named flowspec-map under the BGP flowspec IPv4 address family.

Example

The following example summarizes the commands in this task.

```
device# configure terminal
device(config)# router bgp
device(config-bgp-router)# address-family ipv4 flowspec
device(config-bgp-ipv4fs)# distribute flowspec-map
```

Update a Previously Distributed Route Map

To generate fewer installation requests in the hardware when you make updates to a previously distributed BGP flowspec route map, you can configure a delay period before applying the updates.

About This Task

By default, updates to a previously distributed BGP flowspec route map are applied after a delay of 10 seconds. To send fewer changes to the hardware when making multiple updates, you can configure a longer delay.

The following task shows how to configure a delay of 500 seconds and updates a selection of rules. Your configuration may involve different combinations of match criteria and traffic filtering actions.

Procedure

1. From privileged EXEC mode, enter global configuration mode.

   ```
   device# configure terminal
   ```

2. Configure the delay period that should elapse before application of changes to BGP flowspec route-map rules.

   ```
   device(config)# filter-change-update-delay 500
   ```

   This example configures a delay period of 500 seconds.

3. Enter configuration mode for a sequence number in a previously distributed BGP flowspec route map.

   ```
   device(config)# route-map flowspec_map permit 56
   ```

   This example enters configuration mode for sequence number 56 under a route map named flowspec_map.

4. Configure matching based on the source address.

   ```
   device(config-route-map-flowspec_map/permit/56)# match ipv4 source-address 10.3.2.0/24
   ```

5. Configure matching based on port number.

   ```
   device(config-route-map-flowspec_map/permit/56)# match port lt 40
   ```

   This example specifies that matching occurs when the source or destination port number is less than 40.

   ```
   device(config-route-map-flowspec_map/permit/56)# match port neq 30 67 89
   ```

   This example specifies that matching occurs when the source or destination port number is not equal to 30, 67, and 89.

   When both of these configurations are applied, matching occurs when the source or destination port number is less than 40 or is not equal to 30 and 67 and 89.

6. Configure matching based on fragment type.

```
device(config-route-map-flowspec_map/permit/56)# match fragment-type all first-fragment
```

This example specifies that matching occurs when the traffic fragment type is **first-fragment**.

```
device(config-route-map-flowspec_map/permit/56)# match fragment-type any dont-fragment
not-last-fragment
```

This example specifies that matching occurs when the traffic fragment type is either **dont-fragment** or **not-last-fragment**.

When both of these configurations are applied, matching occurs when the fragment type is equal to **first-fragment**, or **dont-fragment**, or **not-last-fragment**.

7. Enable traffic sampling.

```
device(config-route-map-flowspec_map/permit/56)# set sflow
```

8. Configure a nexthop address.

```
device(config-route-map-flowspec_map/permit/56)# set ip nexthop 10.89.89.7
```

9. Return to privileged EXEC mode.

```
device(config-route-map-flowspec_map/permit/56)# end
```

10. Revert the delay to the default of 10 seconds.

```
device# clear filter-change-update
```

You can revert to the default when, for example, you do not want a long delay because you are installing changes in the hardware.

Example

The following example summarizes the commands in this task.

```
device# configure terminal
device(config)# route-map flowspec_map permit 56# filter-change-update-delay 500
device(config)# route-map flowspec_map permit 56# route-map flowspec_map permit 56
device(config-route-map-flowspec_map/permit/56)# match ipv4 source-address 10.3.2.0/24
device(config-route-map-flowspec_map/permit/56)# match port neq 30 67 89
device(config-route-map-flowspec_map/permit/56)# match port lt 40
device(config-route-map-flowspec_map/permit/56)# match fragment-type all first-fragment
device(config-route-map-flowspec_map/permit/56)# match fragment-type any dont-fragment
not-last-fragment
device(config-route-map-flowspec_map/permit/56)# set set sflow
device(config-route-map-flowspec_map/permit/56)# set ip nexthop 10.89.89.7
device(config-route-map-flowspec_map/permit/56)# end
device# clear filter-change-update
```

Enabling statistics for BGP flowspec rules

By default, BGP flowspec statistics are disabled.

About This Task
Perform the following steps to enable BGP flowspec statistics.

Procedure

1. From privileged EXEC mode, enter global configuration mode.

```
device# configure terminal
```

2. Enable BGP flowspec statistics.

```
device(config)# ip flowspec rules statistics
```

3. Return to global configuration mode.

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

4. Statistics for BGP flowspec can be displayed by using the **show ip flowspec rules detail** command.

```
device# show ip flowspec rules detail

VRF :default-vrf
Total number of Rules: 1

1 Origin: Local(flowmap:23) Active: Yes
  Match:
     DSCP          <60
  Actions:
     Traffic-rate asn 666, rate 125000 bits/sec(operational-rate 132000 bits/sec)

Statistic          packets/bytes
-------------      -----------
   Matched            17412786/12589441782
   Transmitted        1453/1048023
   dropped            17411333/12588393759
```

Example

The following example shows how to enable BGP flowspec statistics.

```
device# configure terminal
device(config)# ip flowspec rules statistics
```

Displaying and clearing statistics for IPv4 flowspec rules

You can display and clear statistical information for IPv4 flowspec rules.

Before You Begin

Before displaying or clearing statistical information for IPv4 flowspec rules, you should enable the collection of statistical information by using the **ip flowspec rules statistics** command. There is a configuration example at the end of this task that shows all the configuration steps in order.

About This Task

Perform the following task to display and then clear statistics for IPv4 flowspec rules.

Procedure

1. From privileged EXEC mode, display statistics for IPv4 flowspec rules. The following example displays detailed statistical information.

```
device# show ip flowspec rules detail

VRF :default-vrf
Total number of Rules: 1

1 Origin: Local(flowmap:23) Active: Yes
  Match:
     DSCP          <60
  Actions:
```

```
      Traffic-rate asn 666,rate 125000 bits/sec(operational-rate 132000 bits/sec)

   Statistic       packets/bytes
   -------------   ------------
     Matched         17412786/12589441782
     Transmitted     1453/1048023
     dropped         17411333/12588393759
```

2. Clear statistics for IPv4 flowspec rules.

```
device# clear ip flowspec rules statistics
```

3. Verify that statistics for IPv4 flowspec rules are cleared.

```
device# show ip flowspec rules detail

VRF :default-vrf
Total number of Rules: 1

1 Origin: Local(flowmap:23) Active: Yes
  Match:
      DSCP         <60
  Actions:
      Traffic-rate asn 666,rate 125000 bits/sec(operational-rate 132000 bits/sec)

   Statistic       packets/bytes
   -------------   ------------
     Matched         0/0
     Transmitted     0/0
     dropped         0/0
```

Example

The following example shows how to enable BGP flowspec rules, display statistics for BGP flowspec rules, clear the statistical information and then verify that the statistics have been cleared.

```
device# configure terminal
device(config)# ip flowspec rules statistics
device(config)# exit
!
device# show ip flowspec rules detail

VRF :default-vrf
Total number of Rules: 1

1 Origin: Local(flowmap:23) Active: Yes
  Match:
      DSCP         <60
  Actions:
      Traffic-rate asn 666,rate 125000 bits/sec(operational-rate 132000 bits/sec)

   Statistic       packets/bytes
   -------------   ------------
     Matched         17412786/12589441782
     Transmitted     1453/1048023
     dropped         17411333/12588393759


!
device# clear ip flowspec rules statistics
!
device# show ip flowspec rules detail

VRF :default-vrf
```

```
Total number of Rules: 1

1 Origin: Local(flowmap:23) Active: Yes
  Match:
     DSCP          <60
  Actions:
     Traffic-rate asn 666,rate 125000 bits/sec(operational-rate 132000 bits/sec)

  Statistic        packets/bytes
  ------------     ------------
    Matched        0/0
    Transmitted    0/0
    dropped        0/0
```

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.

> **Note**
>
> Process restart takes precedence over graceful 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.

About This Task

> **Note**
> High availability (HA) requires GR to be enabled.

Procedure

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

   ```
   device# configure terminal
   ```

2. Enter the **router bgp** command to enable BGP routing.

   ```
   device(config)# router bgp
   ```

3. Enter the **local-as** command to configure the autonomous system number (ASN) in which your device resides.

   ```
   device(config-bgp-router)# local-as 1000
   ```

4. Enter the **neighbor** *ip address* **remote-as** command to add a neighbor.

   ```
   device(config-bgp-router)# neighbor 10.11.12.13 remote-as 2
   ```

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 **graceful-restart** command to enable the graceful restart feature.

   ```
   device(config-bgp-ipv4u)# graceful-restart
   ```

7. 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
     ```

Example

The following example enables the GR feature.

```
device# configure terminal
device(config)# 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
```

Example

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)# 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
```

Example

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)# 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
```

Example

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)# 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
```

What to Do Next

Use the `clear ip bgp neighbor` command with the `all` parameter for the changes to the GR parameters to take effect immediately.

BGP4 Graceful Shutdown

BGP4 graceful shutdown automatically reroutes traffic to an alternate path before shutting down a neighbor. Graceful shutdown minimizes traffic loss during a planned shutdown and re-establishment of BGP4 sessions.

BGP protocol is heavily used in the data center solutions where, for resiliency purposes, redundant spine and leaf devices are deployed to reduce the consequence of breakdowns.

In the context of a planned maintenance operation, such as firmware upgrade, software upgrade, line card replacement, SFP replacement, cable replacement or node replacement, you may need to shut down BGP peer links or a complete switch. This action results in traffic disruption until the underlay network converges, even when alternate paths are available.

By configuring BGP graceful shutdown when an alternate path is available, you can reroute traffic on the backup path before the BGP sessions are torn down (Make before Break). BGP graceful shutdown is an automated procedure that reroutes traffic to the alternate path before shutting down the neighbor. The procedure uses BGP attributes such as LOCAL_PREFERENCE, AS_PATH PREPEND and GRACEFUL_SHUTDOWN community to make a route less preferred.

> **Note**
> You can override these parameters by using a route map.

Table 22: Default BGP graceful shutdown attributes

Attribute	Value	Description
LOCAL_PREFERENCE	1	The LOCAL_PREFERENCE attribute is used between iBGP and Confederation peer devices.
GRACEFUL_SHUTDOWN community	0xFFFF0000	The GRACEFUL_SHUTDOWN community attribute defined by IANA.
AS_PATH PREPEND	Prepend local AS 3 times	AS_PATH PREPEND is used between eBGP peers.
GRACEFUL_SHUTDOWN timer	600 sec	Time between initiation of the BGP Graceful Shutdown and the actual shutdown of BGP neighbors on the initiator device.

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 refreshes routes to all graceful shutdown neighbors with updated BGP attributes, such as LOCAL_PREFERENCE for confederations or iBGP, or AS_PATH PREPEND for eBGP. The best path calculation on all graceful shutdown neighbors learned routes is then triggered, making the routes by way of the graceful shutdown initiator less preferable, so that the traffic from graceful shutdown neighbors is re-routed to alternate paths.

However, the path by way of the graceful shutdown initiator is still valid until the graceful shutdown timer expires. After the graceful shutdown timer expires, all the graceful shutdown neighbors are shut

down on the graceful shutdown initiator and all traffic is rerouted to alternate paths. You can now carry the maintenance operation.

> **Note**
> You should wait for the timer to expire before carrying out a maintenance operation. When the graceful shutdown procedure is deactivated while the timer is running, the route refresh using the graceful shutdown parameters aborts, a new route refresh is triggered, and the graceful shutdown timer stops.

When the graceful shutdown initiator has an outbound policy associated with a particular graceful shutdown neighbor, the outbound policy is applied before the graceful shutdown parameters are applied.

When the graceful shutdown neighbor has an inbound policy associated with the graceful shutdown initiator that is modifying the LOCAL_PREFERENCE or AS_PATH attributes, the graceful shutdown parameters become irrelevant. As a best practice, modify the existing inbound policy rule to match the graceful shutdown community attribute and to make the route less preferred. The send community attribute should also be negotiated between the graceful shutdown initiator and the neighbor.

BGP4 graceful shutdown supports high availability (HA). During a restart, the peer is instantly shut down and the graceful shutdown timer is not started.

> **Note**
> The graceful shutdown community attribute should not be considered for aggregation. To prevent the graceful shutdown behavior being applied to the entire BGP aggregate route when an autonomous system (AS) aggregates multiple routes under a cover prefix, the graceful shutdown community should be filtered out of the aggregate route by using the **no match community** command.

This implementation of BGP4 Graceful Shutdown conforms to RFC 6198 and Graceful BGP session shutdown draft-ietf-grow-bgp-gshut-03.

Considerations

- Graceful shutdown is supported only for the default VRF:
  - Graceful shutdown aims to minimize traffic loss, but it may not completely prevent traffic loss.
  - Graceful shutdown requires the existence of an alternate path; therefore, it is implemented toward a core with multiple spines
- Before the graceful shutdown community attribute can be sent, the send community must be negotiated.
- Modifications made to a route map during the graceful-shutdown period are not effective for graceful shutdown advertisement routes. The graceful-shutdown period is set by using the **neighbor graceful-shutdown** command specifying the **graceful-shutdown** parameter.

Configuring graceful shutdown for all BGP4 neighbors

Configuring graceful shutdown for all BPG4 neighbors minimizes traffic loss during a planned shutdown of all neighbors by rerouting traffic to alternate paths before the shutdown.

About This Task

Perform the following steps to configure graceful shutdown for all BGP4 neighbors.

Procedure

1. From privileged EXEC mode, enter global configuration mode.

   ```
   device# configure terminal
   ```

2. Enable BGP routing.

   ```
   device(config-rbridge-id-122)# router bgp
   ```

3. Configure the autonomous system number (ASN) in which your device resides.

   ```
   device(config-bgp-router)# local-as 1000
   ```

4. Add a neighbor.

   ```
   device(config-bgp-router)# neighbor 10.11.12.13 remote-as 2
   ```

   Repeat this step for all neighbors on the device.

5. Configure graceful shutdown for all neighbors.

   ```
   device(config-bgp-router)# graceful-shutdown 600 community 10
   ```

   This example configures graceful shutdown for all neighbors after 600 seconds and sets the community attribute to 10. The **graceful-shutdown** command gracefully shuts down all neighbors on the node. When a route map is not specified, default graceful shutdown parameters are applied. When a route map is specified, only route-map parameters are applied.

6. Return to privileged EXEC mode.

   ```
   device(config-bgp-router)# end
   ```

7. Verify the configuration.

   ```
   device# show running-configuration router bgp

    router bgp
     local-as 1000
     graceful-shutdown 600 community 10
   !
   ! example truncated for brevity
   ```

Example

The following example shows how to configure graceful shutdown of all BGP4 neighbors after 600 seconds. The community attribute is also set to 10.

```
device# configure terminal
device(config)# 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

Configuring graceful shutdown for a BGP4 peer group minimizes traffic loss during a planned shutdown of the peer group by rerouting traffic to an alternate path before the shutdown.

About This Task

Perform the following steps to configure graceful shutdown for all peers in a BGP4 peer group.

Procedure

1. From privileged EXEC mode, enter global configuration mode.

```
device# configure terminal
```

2. Enable BGP routing.

```
device(config-rbridge-id-122)# router bgp
```

3. Configure the autonomous system number (ASN) in which your device resides.

```
device(config-bgp-router)# local-as 1000
```

4. Add a peer group.

```
device(config-bgp-router)# neighbor grp-1 peer-group
```

This example configures a peer group named grp-1.

5. Configure graceful shutdown for the peer group.

```
device(config-bgp-router)# neighbor grp-1 graceful-shutdown 600 community 20
```

This example configures graceful shutdown of the peer group named grp-1 after 600 seconds and sets the community attribute to 20. This command gracefully shuts down all the peers in a peer-group. When a route map is not specified, default graceful shutdown parameters are applied. When a route map is specified, only route-map parameters are applied.

6. Return to privileged EXEC mode.

```
device(config-bgp-router)# end
```

7. Verify the configuration.

```
device# show running-configuration router bgp

 router bgp
  local-as 1000
  neighbor grp-1 peer-group
  neighbor grp-1 graceful-shutdown 600 community 20
 !
 ! example truncated for brevity
```

Example

The following example shows how to configure graceful shutdown of a peer group named grp-1 after 600 seconds. The community attribute is also set to 20.

```
device# configure terminal
device(config)# 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 specific BGP4 neighbor

Configuring graceful shutdown for a specific BGP4 neighbor minimizes traffic loss during a planned shutdown of a single neighbor by rerouting traffic to an alternate path before the shutdown.

About This Task

Perform the following steps to configure graceful shutdown of a link to a specific BGP4 neighbor.

Procedure

1. From privileged EXEC mode, enter global configuration mode.

   ```
   device# configure terminal
   ```

2. Enable BGP routing.

   ```
   device(config)# router bgp
   ```

3. Configure the autonomous system number (ASN) in which your device resides.

   ```
   device(config-bgp-router)# local-as 1000
   ```

4. Add a neighbor.

   ```
   device(config-bgp-router)# neighbor 10.11.12.13 remote-as 2
   ```

5. Configure graceful shutdown of the neighbor.

   ```
   device(config-bgp-router)# neighbor 10.11.22.13 graceful-shutdown 600 community 30
   ```

   This example configures graceful shutdown of the neighbor (10.11.22.13) after 600 seconds and sets the community attribute to 30.

6. Return to privileged EXEC mode.

   ```
   device(config-bgp-router)# end
   ```

7. Verify the configuration.

   ```
   device# show running-configuration router bgp

    router bgp
     local-as 1000
     neighbor 10.11.12.13 remote-as 2
     neighbor 10.11.12.13 graceful-shutdown 600 community 30
   !
   ! example truncated for brevity
   ```

Example

The following example shows how to configure graceful shutdown of a specific BGP4 neighbor (10.11.22.13) after 600 seconds. The community attribute is also set to 30.

```
device# configure terminal
device(config)# 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 10.11.22.13 graceful-shutdown 600 commuity 30
```

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.

Configure Auto Shutdown of BGP Neighbors on Initial Configuration

Follow this procedure to enable auto shutdown of BGP neighbors on initial configuration.

Procedure

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

   ```
   device# configure terminal
   ```

2. Enter the **router bgp** command to enable BGP routing.

   ```
   device(config)# router bgp
   ```

3. Enter the **local-as** command to configure the autonomous system number (ASN) in which your device resides.

   ```
   device(config-bgp-router)# local-as 65520
   ```

4. 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
   ```

Example

This example summarizes the commands in this procedure.

```
device# configure terminal
device(config)# router bgp
device(config-bgp-router)# local-as 65520
device(config-bgp-router)# auto-shutdown-new-neighbors
```

Disable the BGP4 Peer Shutdown State

The automatic 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. Neighbors that you add to a peer group have the shutdown flag enabled by default.

About This Task

You can disable the shutdown flag.

Procedure

1. Access global configuration mode.

   ```
   device# configure terminal
   ```

2. Access BGP configuration mode.

   ```
   device(config)# router bgp
   ```

3. Identify the autonomous system in which your device resides.

   ```
   device(config-bgp-router)# local-as 65520
   ```

4. Disable the peer shutdown state.

   ```
   device(config-bgp-router)# no neighbor 10.1.1.1 shutdown
   ```

   The BGP4 session establishment process begins.

Example

The following example summarizes the commands in this procedure.

```
device# configure terminal
device(config)# router bgp
device(config-bgp-router)# local-as 65520
device(config-bgp-router)# no neighbor 10.1.1.1 shutdown
```

BGP Additional-paths

BGP additional-paths enables the advertisement of multiple paths for the same prefix without new paths implicitly replacing the previous paths. Path diversity is achieved rather than path hiding.

A BGP device generally advertises only its 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 path hiding is an implicit withdrawal behavior, which achieves better scaling but at the cost of path diversity. 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 must wait for BGP control plane convergence to restore traffic.

The following figure shows two types of path hiding:

- Paths P1 and P2 (for prefix P) are advertised to RR1. P1 is advertised from BR1 and P2 is advertised from BR2. RR1 selects P1 as the best path and advertises only P1 to PE.
- Paths X1 and X2 (for prefix X) are advertised to BR4. X1 is advertised from BR3 with a local preference of 100. BR4 also has path X2 with a local preference 50. However, only the best path (X1) is selected. BR3 advertises X1 to the RRs and X2 is suppressed.
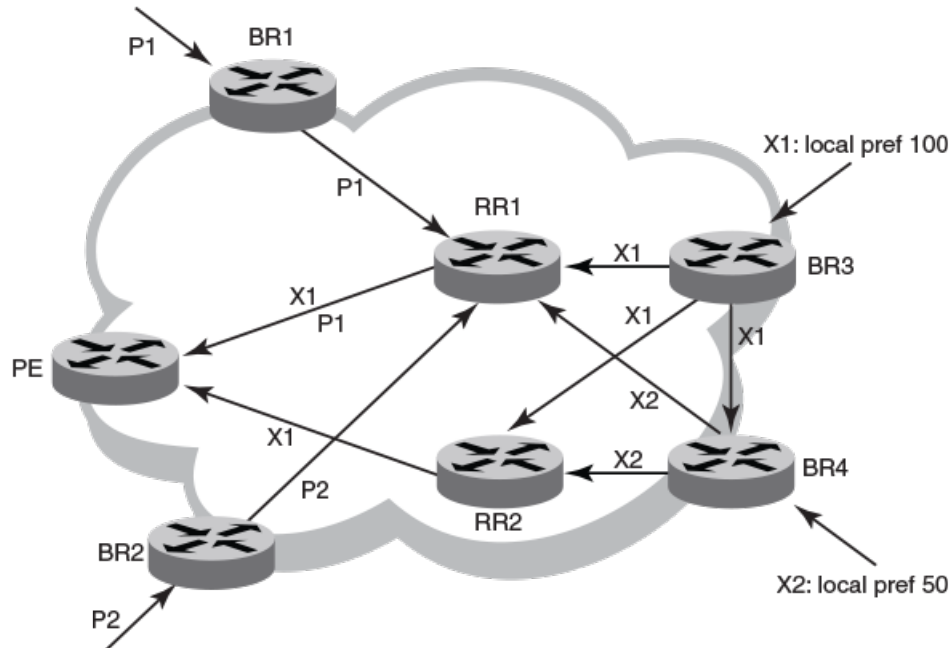
Figure 15: BGP path hiding

BGP additional-paths enables BGP to advertise even secondary best routes, so that multiple paths for the same prefix can be advertised without the new paths implicitly replacing previous paths. BGP additional-paths provides a generic way of offering path diversity.

BGP additional-paths is implemented by including an additional four-octet value known as a path identifier (ID) for each path in the Network Layer Reachability Information (NLRI). Path IDs are unique to a peering session and are generated for each network. That is, a generated path ID is unique for each peer for each prefix.

A BGP device can receive the same path ID for the same prefix from two different peers, or it can receive the same path ID from the same peer for two different prefixes:

- When the same prefix is received with the same path ID from the same peer, it is considered to be a replacement route or duplicate route.
- When the same prefix is received with a different path ID from the same peer, it is considered to be an additional path to the prefix.

To send or receive additional paths, the additional-paths capability must be negotiated. When it is not negotiated, only the best path is sent.

When the additional-paths capability is negotiated, BGP updates carry the path ID. 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 BGP device uses the prefix and path ID to uniquely identify a path advertised to a neighbor so as to continue to send updates for that path. The receiving BGP neighbor that re-advertises a route regenerates its own path ID that is to be associated with the re-advertised route.

The set of additional paths advertised to each neighbor can be different, and advertisement filters are provided for each neighbor.

The following figure shows a BGP additional-paths configuration that supports path diversity.
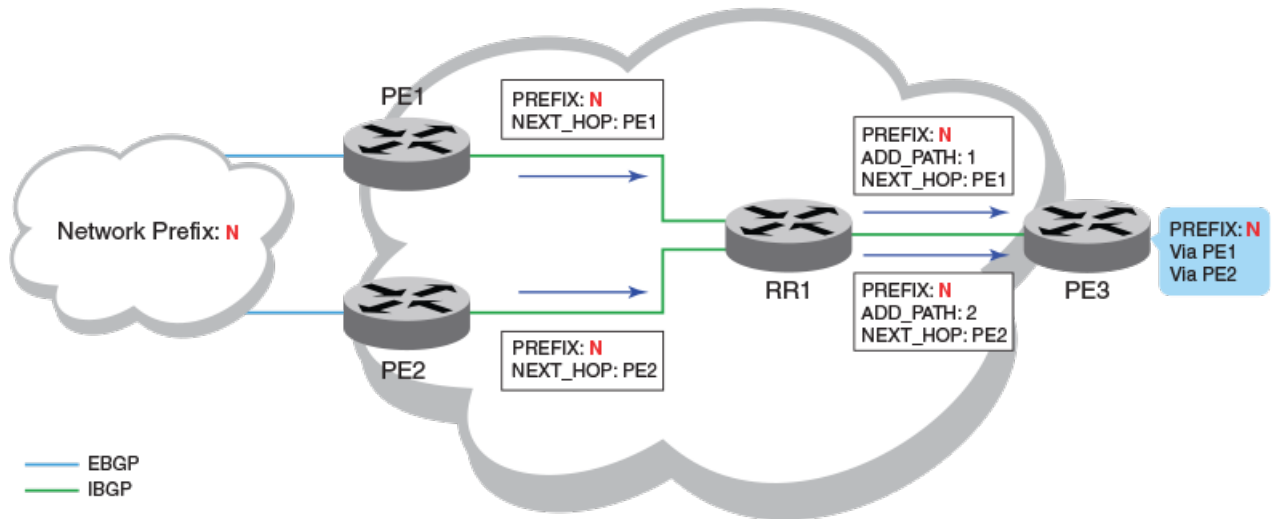
- Although route reflector RR1 has two sources for prefix N, it advertises only the best route to its peers when the additional-paths capability is not configured.
- When the additional-paths capability is configured at RR1 and PE3, PE3 receives two routes to prefix N. Receiving two routes facilitates quick network convergence and recovery:
  - Using PE1 and PE2
  - ECMP paths to PE1 and PE2

Figure 16: BGP additional-paths

Configuration of BGP additional-paths

The configuration of BGP additional-paths involves the following actions:

- **Capability negotiation**: Specifying, at the address family, peer group, or neighbor levels, whether the device can send, receive, or both send and receive additional-paths .
- **Candidate path selection**: Specifying, at the address family level, selection criteria for a set or sets of candidate paths to be advertised.
- **Advertisement of additional-paths from the candidate set**: Specifying, at the neighbor or peer group levels, a set or sets of additional-paths to advertise to a neighbor. The additional-paths to advertise must be from the candidate paths that are marked.

Advantages of BGP additional-paths

- **Fast convergence and fault tolerance**: When BGP additional-paths is enabled, more than one path to a destination is advertised. When one of the paths goes down, connectivity is easily restored due to

the availability of backup paths. When 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 route reflectors (RRs) in an iBGP domain, only the best path is given to clients, even when ECMP paths exist. Giving only the best path affects load balancing. When additional paths are advertised by RRs, the clients have more effective load balancing.

Considerations and limitations for BGP additional-paths

- BGP additional-paths is supported for the following BGP address families:
  - IPv4 unicast
  - IPv4 unicast VRF
  - IPv6 unicast
  - IPv6 unicast VRF
- BGP additional-paths is not supported for BGP VPNv4 unicast, BGP VPNv6, and BGP EVPN address families.
- Changes in the capability of receiving or sending additional-paths are reflected only after the BGP session is restarted.
- RIB-in:
  - When BGP additional-paths is not configured, only one NLRI for each prefix for each peer is supported. Any additional NLRI update for the same prefix from the same peer replaces the existing one.
  - When BGP additional-paths is configured, a device can receive multiple NLRI advertisements for the same prefix from the same peer that are uniquely identified by NLRI path identifiers.
  - The maximum number of additional-paths for each peer for each prefix is 128.
- RIB-out:
  - The maximum number of paths that can be advertised for each prefix is 16. When more than 16 paths for a prefix exist in the RIB-in, only 16 can be advertised.
  - For smooth RIB-out processing, the number of RIB-in paths for any prefix should be maintained within the range from 1 through 16. RIB-out processing time increases exponentially in the scaled scenarios.

Upgrade and downgrade considerations for BGP additional-paths

BGP additional-paths configuration should be removed before a downgrade to a version of software that does not support additional-paths.

When BGP additional-paths is enabled and the configuration saved, an error message occurs when the software is downgraded to an earlier version that does not support the feature. To avoid this error message, the BGP additional-paths configuration should be removed before the downgrade takes place and reconfigured when upgraded again.

Configuring BGP additional-paths and additional-path selection at address-family level

BGP additional-paths is configured by enabling the additional-paths capability and specifying paths that are candidates for selection as additional-paths.

About This Task

BGP additional-paths is supported for the following BGP address families:

- IPv4 unicast
- IPv4 unicast VRF
- IPv6 unicast
- IPv6 unicast VRF

Perform the following task to enable the additional-paths capability and specify paths that are candidates for selection as additional-paths, under the IPv4 unicast address family.

Procedure

1. From privileged EXEC mode, enter global configuration mode.

```
device# configure terminal
```

2. Enable BGP routing.

```
device(config)# router bgp
```

3. Enter configuration mode for the IPv4 unicast address family.

```
device(config-bgp)# address-family ipv4 unicast
```

4. Enable receiving and sending of additional paths.

```
device(config-bgp-ipv4u)# additional-paths receive send
```

A device can be configured to receive, send, or receive and send additional paths.

> **Note**
> A change to the additional-paths capability is activated only after restarting the BGP session.

5. Specify paths that are candidates for selection as additional-paths. The following example specifies that all paths are candidates for selection as additional-paths.

```
device(config-bgp-ipv4u)# additional-paths select all
```

The **all** option configures all (up to a maximum of 16) paths as candidates for selection as additional-paths.

Example

The following example enables the receiving and sending of BGP additional-paths and specifies that all BGP paths are eligible for selection as additional-paths under the IPv4 unicast address family.

```
device# configure terminal
```

```
device(config)# router bgp
device(config-bgp)# address-family ipv4 unicast
device(config-bgp-ipv4u)# additional-paths receive send
device(config-bgp-ipv4u)# additional-paths select all
```

What to Do Next

After completing this task, restart the BGP session to activate the new additional-paths capability
configuration.

Configuring BGP additional-paths and additional-path advertisement at neighbor level

BGP additional-paths can be enabled for a specific peer device (neighbor) under a BGP address family.
In addition, configuration options are available to control the additional-paths that are advertised to
neighbors.

Before You Begin

Before completing this task, the set of paths eligible for selection as additional-paths must be
configured by using the **additional-paths select** command, under the relevant address family.
An example is provided at the end of this task that shows all the configuration steps in order.

About This Task

BGP additional-paths is supported for the following BGP address families:

- IPv4 unicast
- IPv4 unicast VRF
- IPv6 unicast
- IPv6 unicast VRF

Perform the following task to enable additional-paths on a specific peer device and to specify the
additional routes to be advertised by the peer, under the IPv4 unicast address family.

Procedure

1. From privileged EXEC mode, enter global configuration mode.

   ```
   device# configure terminal
   ```

2. Enable BGP routing.

   ```
   device(config)# router bgp
   ```

3. Enter IPv4 unicast address family configuration mode.

   ```
   device(config-bgp)# address-family ipv4 unicast
   ```

4. Enable additional-paths for a specific peer.

```
device(config-bgp-ipv4u)# neighbor 10.60.60.20 additional-paths receive send
```

This example enables the capability to both receive and send additional-paths on peer device
10.60.60.20.

> **Note**
>
> Changes in the capability of receiving or sending additional-paths are reflected only after
> the BGP session is restarted.

5. Configure the additional-paths to be advertised by the peer.

```
device(config-bgp-ipv4u)# neighbor 10.60.60.20 additional-paths advertise all
```

Paths configured for advertisement to neighbors must be a subset of the paths previously
configured by using the **additional-paths select** command under the IPv4 address family.

6. Verify the configuration.

```
device# show ip bgp neighbors

    '+': Data in InQueue '>': Data in OutQueue '-': Clearing
    '*': Update Policy 'c': Group change 'p': Group change Pending
    'r': Restarting 's': Stale '^': Up before Restart '<': EOR waiting

1   IP Address: 10.60.60.20, AS: 200 (IBGP), RouterID: 10.60.60.20, VRF: default-vrf
    State: ESTABLISHED, Time: 4h3m28s, KeepAliveTime: 60, HoldTime: 180
       KeepAliveTimer Expire in 0 seconds, HoldTimer Expire in 159 seconds
    Minimal Route Advertisement Interval: 0 seconds
       RefreshCapability: Received
    Address Family : IPV4 Unicast
       Configured with Add-Path(send receive)capability
       Received Add-Path (send receive)capability in open msg
       Negotiated Add-Path(send receive)capability
    Messages:     Open         Update       KeepAlive   Notification   Refresh-Req
       Sent   : 1            1            275         0              0
       Received: 1            1            275         0              0
    Last Update Time: NLRI       Withdraw         NLRI        Withdraw
                 Tx: 4h3m28s    ---           Rx: 4h3m28s    ---
```

Example

The following example first shows how to configure all paths as eligible for selection as additional-paths
under the IPv4 address family. It then shows how to enable receiving and sending additional-paths on a
specific peer device (10.60.60.20) and configures the advertisement of all paths by 10.60.60.20.

```
device# configure terminal
device(config)# router bgp
device(config-bgp)# address-family ipv4 unicast
device(config-bgp-ipv4u)# additional-paths select all

device(config-bgp-ipv4u)# neighbor 10.60.60.20 additional-paths receive send
device(config-bgp-ipv4u)# neighbor 10.60.60.20 additional-paths advertise all
```

What to Do Next

After completing this task, restart the BGP session to activate the new additional-paths capability
configuration.

Configuring additional-paths and advertisement of additional-paths at peer group level

BGP additional-paths can be enabled for a specific peer group (neighbor) under a BGP address family. In addition, configuration options are available to control the additional-paths that are advertised by the neighbor.

Before You Begin

Before completing this task, configure (by using the **additional-paths select** command) the set of paths eligible for selection as additional-paths under the relevant address family. An example is provided at the end of this task that shows all the configuration steps in order.

About This Task

BGP additional-paths is supported for the following BGP address families:

- IPv4 unicast
- IPv4 unicast VRF
- IPv6 unicast
- IPv6 unicast VRF

Perform the following task to enable additional-paths for a specific peer group and specify the additional routes for the peer group to advertise, under the IPv4 unicast address family.

Procedure

1. From privileged EXEC mode, enter global configuration mode.

   ```
   device# configure terminal
   ```

2. Enable BGP routing.

   ```
   device(config)# router bgp
   ```

3. Create a peer group. The following example shows how to create a peer group named pg-1

   ```
   device(config-bgp)# neighbor pg-1 peer-group
   ```

4. Enter IPv4 unicast address family configuration mode.

   ```
   device(config-bgp)# address-family ipv4 unicast
   ```

5. Enable additional-paths for the peer group.
   ```
   device(config-bgp-ipv4u)# neighbor pg-1 additional-paths receive send
   ```

   This example enables the capability to both receive and send additional-paths on peer group pg-1.

   > **Note**
   > Changes in the capability of receiving or sending additional-paths are reflected only after the BGP session is restarted.

6. Configure the additional-paths to be advertised by the peer group. The following example shows how to configure the advertisement of group-best paths.

```
device(config-bgp-ipv4u)# neighbor pg-1 additional-paths advertise group-best
```

Paths configured for advertisement to neighbors must be a subset of the paths previously configured by using the **additional-paths select** command under the IPv4 address family.

Example

The following example first shows how to configure **group-best** paths as candidates for selection as additional paths, under the IPv4 address family. It then shows how to enable receiving and sending additional-paths for peer group pg-1 and configures the advertisement of group-best paths for pg-1.

```
device# configure terminal
device(config)# router bgp
device(config-bgp)# address-family ipv4 unicast
device(config-bgp-ipv4u)# additional-paths select group-best
device(config-bgp-ipv4u)# exit

device(config-bgp)# neighbor pg-1 peer-group
device(config-bgp)# address-family ipv4 unicast
device(config-bgp-ipv4u)# neighbor pg-1 additional-paths receive send
device(config-bgp-ipv4u)# neighbor pg-1 additional-paths advertise group-best
```

What to Do Next

After completing this task, restart the BGP session to activate the new additional-paths capability configuration.

Filtering additional-paths advertised at route-map level

You can configure a route map instance to filter the additional-paths that are advertised.

Before You Begin

Before completing this task, the set of paths eligible for selection as additional-paths must be configured by using the **additional-paths select** command, under the relevant address family. An example is provided at the end of this task that shows all the configuration steps in order.

About This Task

Perform the following task to configure a route-map instance to filter the additional-paths advertised.

Procedure

1. From privileged EXEC mode, enter global configuration mode.

```
device# configure terminal
```

2. Create a route map instance and enter configuration mode for the route map instance.

```
device(config)# route-map rm-1 permit 123
```

This example creates instance 123 of a route map named rm-1 and enters configuration mode for the instance.

3. Configure filtering of additional-paths advertised for the route-map instance.

```
device(config-route-map/permit/123)# match additional-paths advertise-set best-range 2
13
```

This example configures the route-map instance to advertise paths with a path marking (tag) in the range from 2 through 13.

> **Note**
>
> Only one **match additional-paths advertise-set** configuration is allowed for a route map instance; any subsequent **additional-paths advertise-set** configuration overwrites the previous configuration.

> **Note**
>
> You cannot use a route map that is configured with an additional-paths match statement for incoming routes (routes that are configured by using the **neighbor route-map** command specifying the **in** option.

Example

The following example first shows how to configure all paths to be eligible for selection as additional-paths, under the IPv4 address family. It then configures a route map (rm-1) to advertise paths marked in the range from 2 through 14.

```
device# configure terminal
device(config)# router bgp
device(config-bgp)# address-family ipv4 unicast
device(config-bgp-ipv4u)# additional-paths select all
device(config-bgp-ipv4u)# end

device(config)# route-map rm-1 permit 123
device(config-route-map/permit/123)# match additional-paths advertise-set best-range 2 13
```

Disable BGP Additional-paths for a Neighbor

The additional-paths capability can be disabled for a neighbor or peer group under an address family.

About This Task

When an additional-paths capability is configured for a BGP address family, you can disable the capability for a specific neighbor or peer group under the address family.

Procedure

1. From privileged EXEC mode, enter global configuration mode.

```
device# configure terminal
```

2. Enable BGP routing.

```
device(config)# router bgp
```

3. Enter configuration mode for the IPv4 unicast address family.

```
device(config-bgp)# address-family ipv4 unicast
```

4. Disable the additional-paths capability for a specific peer device.

```
device(config-bgp-ipv4u)# neighbor 10.12.12.12 additional-paths disable
```

This example disables the additional-paths capability for peer device 10.12.12.12

When you run this command for an address-family level on which the additional-paths capability is not configured, an error message is displayed.

5. Restart the BGP session to activate the new additional-paths capability configuration.

Example

This example re-enables the additional-paths capability on the specific neighbor 10.12.12.12 in the IPv4 unicast address family.

```
evice# configure terminal
device(config)# router bgp
device(config-bgp)# address-family ipv4 unicast
device(config-bgp-ipv4u)# no neighbor 10.12.12.12 additional-paths disable
```

BGP best-external route

The best external route is the most preferred route among those received from external neighbors. Advertising the best external route supports fast restoration of connectivity.

The best external route, when different from the best route, acts as a backup route and introduces additional information into an Interior Border Gateway Protocol (iBGP) mesh, which may be useful in restoration of connectivity.

In active-backup topologies, service providers may use routing policies that cause a border router to choose a path received over an iBGP session (of another border router) as the best path for a prefix even when an Exterior Border Gateway Protocol (eBGP) learned path exists on the device. This type of active-backup topology defines one exit or egress point for the prefix in the autonomous system and uses the other points as backups when the primary link or eBGP peering is not available. When advertisement of the best-external path is not configured, such a routing policy causes the border router to hide (from the autonomous system) the paths learned over its eBGP sessions because it does not advertise these paths.

When advertisement of the best external path is configured by using the **best-external** command, the best external learned path, when different from the best route, is advertised and acts as a backup route that supports fast restoration of connectivity.

The following figure illustrates the impact of best-external configuration in a BGP topology.
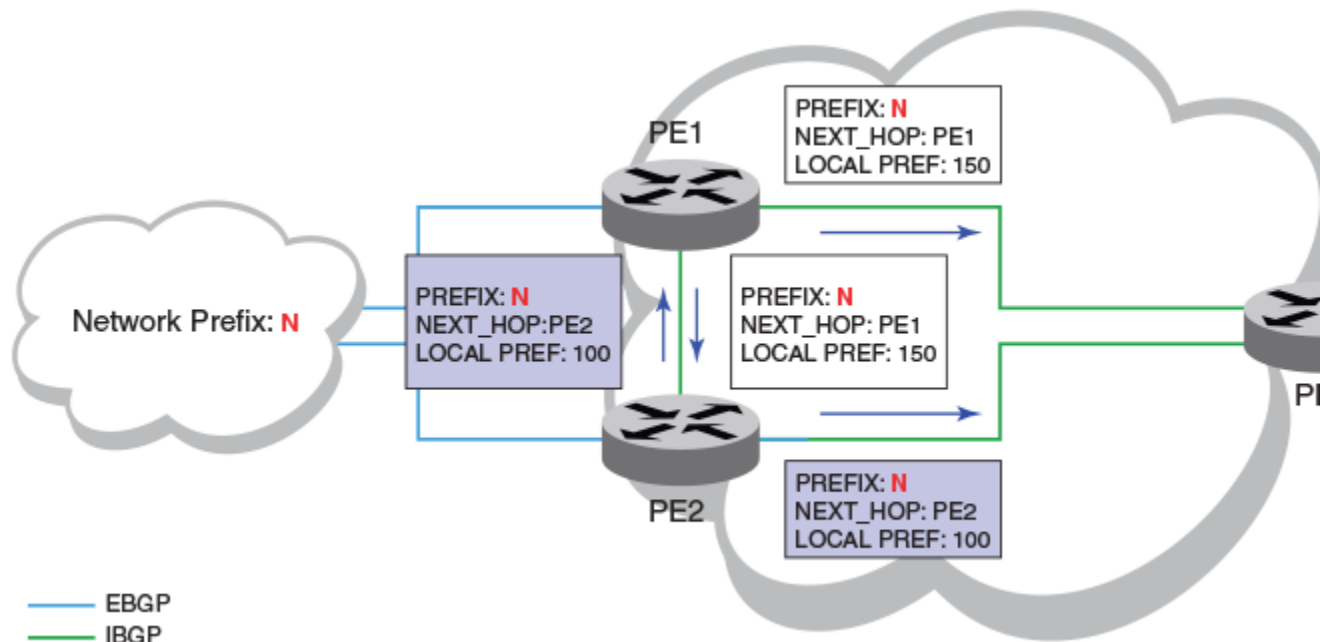
- PE1 is the primary path and PE2 is the backup path to network N
- When advertisement of the best external path is not configured:
  - PE2 does not advertise prefix N to its iBGP peers because PE2 prefers the iBGP route from PE1, which has a higher local preference, as the best route when compared to its best external (eBGP) route
  - PE3 has one path for prefix N
- When advertisement of the best external path is configured:
  - PE2 propagates its best external path to its iBGP peers
  - PE3 has two paths for prefix N

Figure 17: BGP best-external configuration

Configuring BGP best-external route at address-family level

When the best external route is advertised to internal peers, it acts as a backup route that may be useful in the restoration of connectivity.

About This Task

Perform the following task to store the best external route (in addition to the best route) and advertise it to internal peers under the IPv4 unicast address family.

Procedure

1. From privileged EXEC mode, enter global configuration mode.

```
device# configure terminal
```

2. Enable BGP routing.

```
device(config)# router bgp
```

3. Enter IPv4 unicast address family command mode.

```
device(config-bgp)# address-family ipv4 unicast
```

4. Configure the storage and advertisement, to internal peers, of the best external route.

```
device(config-bgp-ipv4u)# advertise-best-external
```

5. Return to privileged EXEC mode.

```
device(config-bgp-ipv4u)# end
```

6. Verify the configuration.

```
device# show ip bgp

Total number of BGP Routes: 4
Status codes: s suppressed, d damped, h history, * valid, > best, i internal, S stale,
x best-external
Origin codes: i - IGP, e - EGP, ? - incomplete
    Network           Next Hop        MED        LocPrf     Weight Path
*>i 110.110.110.0/24  50.50.50.10                150        0      i
*x  110.110.110.0/24  20.20.20.10                100        0      200 i
*   110.110.110.0/24  30.30.30.10                100        0      300 i
*   110.110.110.0/24  40.40.40.10                100        0      400 i
```

Example

The following example shows how to store the best external route and advertise it to internal peers, under the IPv4 unicast address family.

```
device# configure terminal
device(config)# router bgp
device(config-bgp)# address-family ipv4 unicast
device(config-bgp-ipv4u)# advertise-best-external
```

Generalized TTL Security Mechanism

Generalized TTL Security Mechanism (GTSM) is a lightweight security mechanism that protects external Border Gateway Protocol (eBGP) peering sessions from CPU utilization-based attacks that use forged IP packets.

GTSM prevents attempts to hijack the eBGP peering session from the following attackers:

- A host on a network segment that is not part of either BGP network
- A host on a network segment that is not between the eBGP peers

You enable GTSM by configuring a minimum Time To Live (TTL) value for IP packets incoming 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 for the peering session. If the TTL value in the packet header is less than the value for the peering session, the packet is silently discarded and no Internet Control Message Protocol (ICMP) message is generated.

For directly connected neighbors, the device expects the BGP control packets from the neighbor to have a TTL value of 254 or 255. For multihop peers, the device expects the TTL for BGP control packets from the neighbor to be greater than or equal to 255, minus the configured number of hops to the

neighbor. The device drops the BGP control packets from the neighbor if the packets do not have the expected value.

Considerations

- GTSM is supported for directly connected peering sessions and for 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.

Configure GTSM for BGP4

Generalized TTL Security Mechanism (GTSM) can be configured to protect external Border Gateway Protocol (eBGP) peering sessions.

Procedure

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

   ```
   device# configure terminal
   ```

2. Enter the **router bgp** command to enable BGP routing.

   ```
   device(config)# router bgp
   ```

3. Enter the **local-as** command to configure the autonomous system number (ASN) in which your device resides.

   ```
   device(config-bgp-router)# local-as 65520
   ```

4. Enter the **neighbor remote-as** command to add a neighbor.

   ```
   device(config-bgp-router)# neighbor 10.10.10.1 remote-as 2
   ```

5. Enter the **neighbor ebgp-btsh** command, specifying an IP address, to enable GTSM.

   ```
   device(config-bgp-router)# neighbor 10.10.10.1 ebgp-btsh
   ```

Example

This example summarizes the commands in this procedure.

```
device# configure terminal
device(config)# 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.

Procedure

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

   ```
   device# configure terminal
   ```

2. Enter the **router bgp** command to enable BGP routing.

   ```
   device(config)# router bgp
   ```

3. Enter the **address-family ipv4 unicast** command to enter BGP IPv4 address family unicast configuration mode.

   ```
   device(config-bgp-router)# address-family ipv4 unicast
   ```

4. 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
   ```

Example

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)# router bgp
device(config-bgp-router)# address-family ipv4 unicast
device(config-bgp-ipv4u)# neighbor 10.1.1.1 allowas-in 3
```

BGP4 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 changes in route states 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.

The **dampening** command offers several parameters that you can use to enable and fine-tune the route dampening feature.

half-life	The number of minutes after which the route penalty becomes half its value. The default is 15.
reuse	Minimum penalty below which the route becomes usable again. The default is 750.
suppress	Maximum penalty above which the route is suppressed by the device. The default is 2000.
max-suppress-time	Maximum number of minutes that a route can be suppressed by the device. The default is 40.

Enable BGP4 Route Dampening

You can enable the default dampening parameters or specify the values you want.

Procedure

1. Access global configuration mode.

   ```
   device# configure terminal
   ```

2. Access BGP configuration mode.

   ```
   device(config)# router bgp
   ```

3. Access address-family ipv4 unicast configuration mode.

   ```
   device(config-bgp-router)# address-family ipv4 unicast
   ```

4. Enable the default dampening parameter values.

   ```
   device(config-bgp-ipv4u)# dampening
   ```

Example

This example sets the half-life parameter to 20, the reuse parameter to 200, the suppress parameter to 2500, and the max-suppress-time parameter to 40.

```
device# configure terminal
device(config)# router bgp
device(config-bgp-router)# address-family ipv4 unicast
device(config-bgp-ipv4u)# dampening 20 200 2500 40
```

Configure the BGP4 Route Dampening Penalty with a Route Map

You can set a BGP route-flap dampening penalty in a route-map instance..

Procedure

1. Access global configuration mode.

   ```
   device# configure terminal
   ```

2. Create a route map instance and allow a matching pattern.

   ```
   device(config)# route-map myroutemap permit 1
   ```

   This example shows the **route-map** command, the route map name (myroutemap), the **permit** keyword, and the matching pattern of 1.

3. Specify the dampening penalty for the route-map instance.

```
device(config-route-map-myroutemap/permit/1)# set dampening 20
```

This example specifies a penalty of 20.

Reactivate Dampened BGP4 Routes

You can use a clear command to reactivate all dampened routes

Procedure

1. Access privileged EXEC mode.
2. Display all dampened BGP4 routes to verify that you want to reactivate them.

```
device# show ip bgp dampened-paths

    Status Code  >:best d:damped h:history *:valid
    Network            From           Flaps Since       Reuse       Path
*d  110.110.114.0/24   160.160.160.10  38    0 :3 :49    0 :10:10    111
*d  110.110.113.0/24   160.160.160.10  38    0 :3 :49    0 :10:10    111
*d  110.110.112.0/24   160.160.160.10  38    0 :3 :49    0 :10:10    111
*d  110.110.111.0/24   160.160.160.10  38    0 :3 :49    0 :10:10    111
*d  110.110.110.0/24   160.160.160.10  38    0 :3 :49    0 :10:10    111
```

3. Reactivate all dampened BGP4 routes.

```
device# clear ip bgp dampening
```

4. Verify that there are no dampened BGP4 routes.

```
device# show ip bgp dampened-paths
device#
```

BGP4 Diagnostic Buffers

The device stores BGP4 diagnostic information in buffers.

The device stores the following information:

* The first 400 bytes of the last received packet that contained an error
* The last NOTIFICATION message sent or received by the device

This diagnostic 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.

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 one neighbor, or for all neighbors in a specific peer group.

Display BGP4 Statistics

Various **`show ip bgp`** commands verify information about BGP4 configurations.

About This Task

Use one or more of the following commands to verify BGP4 information. The commands do not have to be entered in this order.

Procedure

1. Enter the **`show ip bgp summary`** command.

```
device# show ip bgp summary

  BGP4 Summary
  Router ID: 192.117.117.36   Local AS Number: 1001
  Confederation Identifier: not configured
  Confederation Peers:
  Maximum Number of IP ECMP Paths Supported for Load Sharing: 1
  Number of Neighbors Configured: 1, UP: 0
  Number of Routes Installed: 4, Uses 384 bytes
  Number of Routes Advertising to All Neighbors: 4 (4 entries), Uses 240 bytes
  Number of Attribute Entries Installed: 1, Uses 104 bytes
  Neighbor Address  AS#        State     Time     Rt:Accepted Filtered Sent    ToSend
  192.117.117.3    1001        ACTIV    4h 5m 0s   0          0        0       4
```

This example output gives summarized BGP4 information.

2. Enter the **`show ip bgp routes`** command.

```
device# show ip bgp routes

Total number of BGP Routes: 4
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      42.9.1.0/24     0.0.0.0        0          100        32768  BL
         AS_PATH:
2      42.9.2.0/24     0.0.0.0        0          100        32768  BL
         AS_PATH:
3      42.9.3.0/24     0.0.0.0        0          100        32768  BL
         AS_PATH:
4      42.9.4.0/24     0.0.0.0        0          100        32768  BL
         AS_PATH:
```

This example shows general BGP4 route information.

3. Enter the **`show ip bgp`** command.

```
device# show ip bgp

Total number of BGP Routes: 4
Status codes: s suppressed, d damped, h history, * valid, > best, i internal, S stale
Origin codes: i - IGP, e - EGP, ? - incomplete
    Network          Next Hop       MED        LocPrf     Weight Path
*>  42.9.1.0/24      0.0.0.0        0          100        32768  i
*>  42.9.2.0/24      0.0.0.0        0          100        32768  i
*>  42.9.3.0/24      0.0.0.0        0          100        32768  i
*>  42.9.4.0/24      0.0.0.0        0          100        32768  i
```

This example shows general BGP4 information.

4. Enter the **show ip bgp attribute-entries** command.

```
device# show ip bgp attribute-entries

        Total number of BGP Attribute Entries: 1
1       Next Hop  : 0.0.0.0              MED       :0                  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: 0x67c1002c  Hash:364 (0x01000000)
        Links: 0x00000000, 0x00000000
        Reference Counts: 4:0:4, Magic: 1
```

This example shows information about one route-attribute entry that is stored in device memory.

5. Enter the **show ip bgp peer-group** command.

```
device# show ip bgp peer-group

1   BGP peer-group is pg
       Address family : IPV4 Unicast
         activate
       Address family : IPV6 Unicast
         no activate
    Members:
        IP Address: 1.1.1.1, AS: 100
        IP Address: 1::1, AS: 100

2   BGP peer-group is pg6
       Address family : IPV4 Unicast
         activate
       Address family : IPV6 Unicast
         no activate
    Currently there are no members
```

This example shows output for two BGP peer groups, called "pg" and "pg6".

6. Enter the **show ip bgp routes** command using the **summary** keyword.

```
device# show ip bgp routes summary

  Total number of BGP routes (NLRIs) Installed    : 1
  Distinct BGP destination networks               : 1
  Filtered bgp routes for soft reconfig           : 0
  Routes originated by this router                : 1
  Routes selected as BEST routes                  : 1
  Routes Installed as BEST routes                 : 1
  BEST routes not installed in IP forwarding table : 0
  Unreachable routes (no IGP route for NEXTHOP)   : 0
  IBGP routes selected as best routes             : 0
  EBGP routes selected as best routes             : 0
  BEST routes not valid for IP forwarding table   : 0
```

This example shows summarized BGP4 route information.

Display BGP4 Neighbor Statistics

Various **show ip bgp neighbor** commands verify information about BGP4 neighbor configurations.

About This Task

Use one or more of the following commands to verify BGP4 neighbor information. The commands do not have to be entered in this order.

Procedure

1. Enter the **show ip bgp neighbors** command.

```
device# show ip bgp neighbors

   Total number of BGP Neighbors: 2
1   IP Address: 123.123.123.3, AS: 333 (EBGP), RouterID: 9.9.9.9, VRF: default-vrf
    State: ESTABLISHED, Time: 0h1m32s, KeepAliveTime: 60, HoldTime: 180
       KeepAliveTimer Expire in 17 seconds, HoldTimer Expire in 147 seconds
    Minimal Route Advertisement Interval: 0 seconds
       RefreshCapability: Received
    Messages:    Open    Update  KeepAlive Notification Refresh-Req
       Sent   : 2       15      3339      1           0
       Received: 2      0       3356      0           0
    Last Update Time: NLRI              Withdraw              NLRI
Withdraw
                Tx: 0h1m32s            ---                   Rx: ---
---
    Last Connection Reset Reason:User Reset Peer Session
    Notification Sent:    Cease/Administrative Reset
    Notification Received: Unspecified
    Neighbor NLRI Negotiation:
      Peer configured for IPV4 unicast  Routes
    Neighbor ipv6 MPLS Label Capability Negotiation:
    Neighbor AS4 Capability Negotiation:
    Outbound Policy Group:
       ID: 2, Use Count: 2
    BFD:Disabled
       Byte Sent:   146, Received: 0
       Local host:  123.123.123.2, Local  Port: 44575
       Remote host: 123.123.123.3, Remote Port: 179

2   IP Address: 160.160.160.10, AS: 111 (EBGP), RouterID: 193.24.0.1, VRF: default-vrf
    State: ESTABLISHED, Time: 0h1m33s, KeepAliveTime: 30, HoldTime: 90
       KeepAliveTimer Expire in 12 seconds, HoldTimer Expire in 86 seconds
    Minimal Route Advertisement Interval: 0 seconds
       RefreshCapability: Received
    Messages:    Open    Update  KeepAlive Notification Refresh-Req
       Sent   : 8       0       553       5           0
       Received: 8      9       498       0           0
    Last Update Time: NLRI              Withdraw              NLRI
Withdraw
                Tx: ---               ---                   Rx: 0h1m33s
---
    Last Connection Reset Reason:User Reset Peer Session
    Notification Sent:    Cease/Administrative Reset
    Notification Received: Unspecified
    Neighbor NLRI Negotiation:
      Peer configured for IPV4 unicast  Routes
    Neighbor ipv6 MPLS Label Capability Negotiation:
    Neighbor AS4 Capability Negotiation:
    Outbound Policy Group:
       ID: 2, Use Count: 2
```

```
        BFD:Disabled
          Byte Sent:   121, Received: 0
          Local host:  160.160.160.20, Local  Port: 53791
          Remote host: 160.160.160.10, Remote Port: 179
```

This example output gives general information about BGP4 neighbors.

2. Enter the **show ip bgp neighbors advertised-routes** command.

```
device# show ip bgp neighbors 123.123.123.3 advertised-routes

        There are 5 routes advertised to neighbor 123.123.123.3
Status A:AGGREGATE B:BEST b:NOT-INSTALLED-BEST E:EBGP I:IBGP L:LOCAL
        Prefix              Next Hop          MED         LocPrf      Weight Status
1       110.110.110.0/24    123.123.123.2     0                       0      BE
          AS_PATH: 222 111
2       110.110.111.0/24    123.123.123.2     0                       0      BE
          AS_PATH: 222 111
3       110.110.112.0/24    123.123.123.2     0                       0      BE
          AS_PATH: 222 111
4       110.110.113.0/24    123.123.123.2     0                       0      BE
          AS_PATH: 222 111
5       110.110.114.0/24    123.123.123.2     0                       0      BE
          AS_PATH: 222 111
```

This example shows information about all the routes the BGP4 networking device advertised to the neighbor.

3. Enter the **show ip bgp neighbors last-packet-with-error** command.

```
device# show ip bgp neighbors 123.123.123.3 last-packet-with-error

 Received Message Length: 45
 BGP Message:
  0xffffffff  0xffffffff  0xffffffff  0xffffffff  0x002d0104
  0x014b00b4  0x09090909  0x10020601  0x04010000  0x01020202
  0x00020280  0x00

 BGP Header
  Marker:   0xffffffff  0xffffffff  0xffffffff  0xffffffff
  Message Length: (0x002d) 45
  Message Type: (0x01) OPEN

 OPEN Message
 Version: (0x04) 4
 AS Number: (0x014b) 331
 Hold Time: (0x00b4) 180
 BGP Identifier: (0x09090909) 9.9.9.9
 Optional Parameter length: (0x10) 16

 OPEN message optional parameters
  Parameter Type: (0x02) Capability
  Parameter Length: (0x06) 6
   Capability Type: (0x01) MULTIPROTOCOL EXTENSIONS
   Capability Length: (0x04) 4
   AFI: (0x0100) Unknown(256)
   Reserved: (0x00) 0
   SAFI: (0x01) Unicast

  Parameter Type: (0x02) Capability
  Parameter Length: (0x02) 2
   Capability Type: (0x02) ROUTE REFRESH(new)
   Capability Length: (0x00) 0

  Parameter Type: (0x02) Capability
```

```
   Parameter Length: (0x02) 2
    Capability Type: (0x80) ROUTE REFRESH(old)
    Capability Length: (0x00) 0
```

This example shows information about the last packet that contained an error from any of a device's neighbors.

4. Enter the **show ip bgp neighbors received-routes** command.

```
device# show ip bgp neighbors 160.160.160.10 received-routes

        There are 5 received routes from neighbor 160.160.160.10
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       110.110.110.0/24    160.160.160.10  0         100        0      BE
          AS_PATH: 111
2       110.110.111.0/24    160.160.160.10  0         100        0      BE
          AS_PATH: 111
3       110.110.112.0/24    160.160.160.10  0         100        0      BE
          AS_PATH: 111
4       110.110.113.0/24    160.160.160.10  0         100        0      BE
          AS_PATH: 111
5       110.110.114.0/24    160.160.160.10  0         100        0      BE
          AS_PATH: 111
```

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 ip bgp neighbors rib-out-routes** command.

```
device# show ip bgp neighbors 123.123.123.3 rib-out-routes

        There are 5 RIB_out routes for neighbor 123.123.123.3
Status A:AGGREGATE B:BEST b:NOT-INSTALLED-BEST E:EBGP I:IBGP L:LOCAL
        Prefix              Next Hop        MED       LocPrf     Weight Status
1       110.110.110.0/24    160.160.160.10  0         100        0      BE
          AS_PATH: 111
2       110.110.111.0/24    160.160.160.10  0         100        0      BE
          AS_PATH: 111
3       110.110.112.0/24    160.160.160.10  0         100        0      BE
          AS_PATH: 111
4       110.110.113.0/24    160.160.160.10  0         100        0      BE
          AS_PATH: 111
5       110.110.114.0/24    160.160.160.10  0         100        0      BE
          AS_PATH: 111
```

This example shows information about BGP4 outbound RIB routes.

BGP PIC

Normal BGP-based convergence can take minutes (depending on the number of prefixes) to complete. BGP Prefix-Independent Convergence (PIC) is an IETF standards-based method that accelerates data path convergence (to sub-seconds) under failover conditions. PIC is a data-plane feature that does not affect the control plane.

Functional overview

BGP registers with the routing information base manager (RIBM) to resolve a BGP next-hop, which in turn may be reachable through an Interior Gateway Protocol (IGP) prefix. Ultimately, BGP uses the IGP

next-hop (instead of the BGP next-hop) to install the prefixes in the RIBM and the forwarding information base (FIB). Historically, this scheme worked well for small routing table sizes and was considered advantageous for data-plane implementations because the number of lookups required is constant and results in higher throughput. In the context of BGP PIC, such an implementation is loosely referred to as a "flat" (not hierarchical) RIBM and FIB implementation.

Continuing with the flat implementation scenario, assume that because of a change in network topology, the BGP next-hop reachability information changes. In this situation, when the RIBM informs BGP about the change, BGP reruns the best route calculations for the affected prefixes, which can take a while depending on the number of prefixes. During this period, data traffic is directed to an offline or disconnected router (black-holed) until the route calculations are complete. In summary, convergence time becomes proportional to the number of prefixes.

As the Internet routing table grows along with the number of prefixes in a border router scenario, the flat approach makes convergence very slow. BGP PIC was designed to handle network outages in a prefix-independent way.

The BGP PIC standard handles data plane disruption due to network core failures (node or link) with the hierarchical RIBM and FIB approach. However, for network edge failures (node or link), BGP PIC requires additional path (ECMP or backup) support from BGP to achieve the same level of performance.

Hierarchical RIBM and FIB

BGP provides prefixes that point to next-hops, which are reachable through the IGP prefix and are ultimately followed by the IGP next-hop. This hierarchy is maintained in the RIBM and FIB.

When an IGP topology change occurs in the network core, the RIBM and FIB identify the affected BGP next-hops and update the next-hop adjacency information in the hardware without affecting the already programmed prefixes. Because the number of next-hops is limited in a network, the time required to update the data plane is sub-seconds with BGP PIC.

SLX-OS does not maintain the forwarding hierarchy in hardware tables. Instead, the hierarchical mapping is maintained only at the RIBM and FIB level. When notified by IGP (or BGP by means of BFD) of a topology change, the RIBM and FIB issue next-hop updates that change the next-hop hardware table. This design may require more time for the data plane to converge, but it provides a good trade-off between performance and scale.

BGP additional paths

In most commonly followed designs, BGP provides only the best path to the RIBM for route installation.

If the chosen path is an ECMP path and one of the next-hops is not reachable, with the help of the hierarchical RIBM and FIB the affected next-hop can be easily removed from ECMP at the RIBM and FIB level and at the hardware level.

However, when BGP cannot provide an ECMP next-hop, and if faster data-plane convergence is expected in a prefix-independent manner, BGP PIC recommends that BGP provide primary and backup paths to the RIB. The backup path is not programmed into hardware and is maintained only at the RIBM and FIB level.

When network changes affect BGP primary next-hop reachability, the RIBM and FIB can identify the affected BGP next-hop and switch to the backup next-hop.

BGP additional paths are supported by BGP PIC, with the following well-known schemes:

- Full mesh iBGP (without a route reflector)
- Add path
- BGP best external

Supported network triggers for failover

The following network triggers cause PIC to restore traffic within sub-seconds.

- BGP notification of triggers:
  - BFD notification of neighbor down
  - Execution of **neighbor shutdown** on a remote peer, causing the session to be down
  - Execution of **clear ip bgp neighbor** on a remote peer
  - Removal of the configuration on a remote peer
- IGP notification of a topology change
- Interface state going down

BGP PIC functional scenarios

BGP PIC functionality varies based on the location and type of network failure.

BGP PIC core node failure

The following figure illustrates a BGP PIC core node failure and the behavior of the route tables.
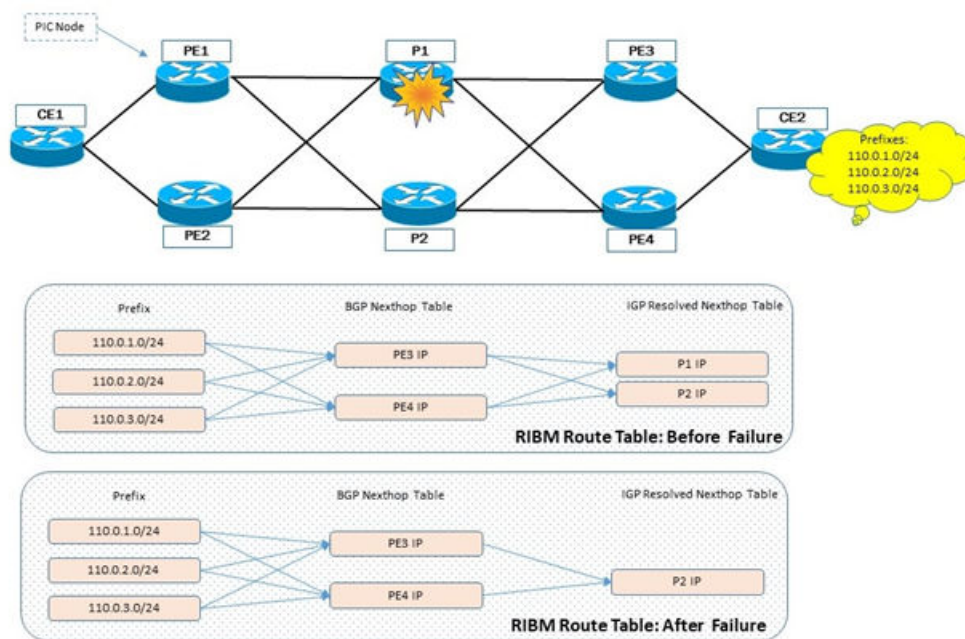
Figure 18: BGP PIC core node failure

The behavior is as follows:

1. At PE1, there are two paths to each CE2 prefix, through PE3 and PE4.
2. BGP next-hop PE3 is reachable through P1 and P2. The same applies to next-hop PE4.
3. When P1 fails, IGP invalidates the path to PE3 through P1. PE3 is now reachable only through P2.
4. The RIBM and FIB perform a reverse lookup to determine which BGP next-hop is affected. Because the number of BGP next-hops is usually less than the number of prefixes, the lookup time is reduced.
5. After the affected BGP next-hops are identified, the RIBM and FIB issue a next-hop update to the hardware SDK to use the updated list of IGP-resolved next-hops.

In the Figure 18 on page 213 example, a next-hop update is issued for BGP next-hops PE3 and PE4.

BGP PIC core link failure

The following figure illustrates a BGP PIC core link failure and the behavior of the route tables.
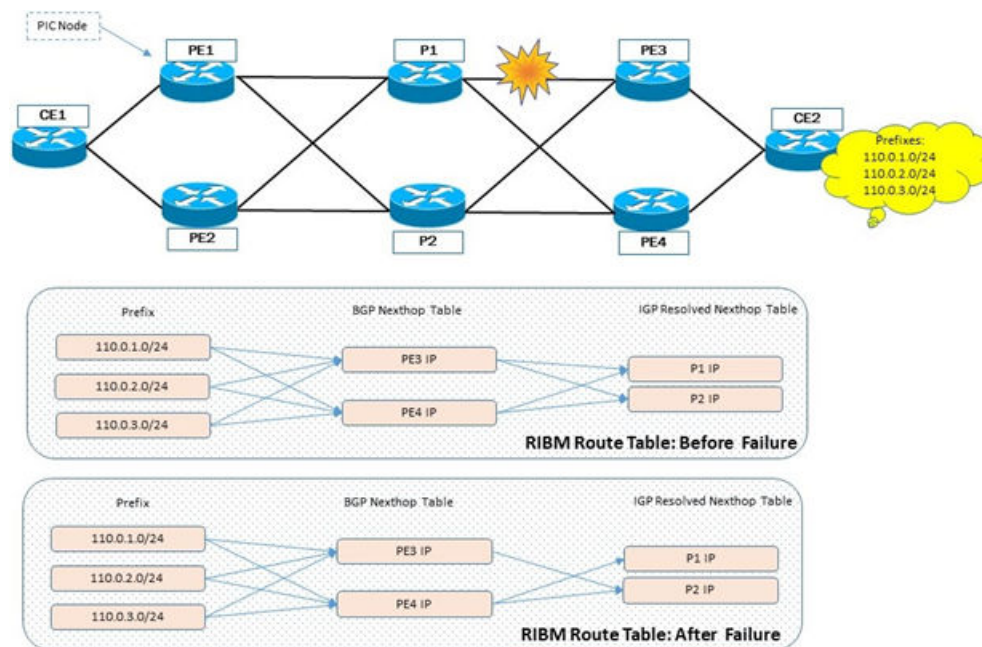
Figure 19: BGP PIC core node failure

The behavior is as follows:

1. At PE1, there are two paths to each CE2 prefix, through PE3 and PE4.
2. BGP next-hop PE3 is reachable through P1 and P2. The same applies to next-hop PE4.
3. Following a P1-to-PE3 link failure, IGP invalidates the path to PE3 through P1. PE3 is now reachable only through P2.
4. The RIBM and FIB perform a reverse lookup to determine which BGP next-hop is affected. Because the number of BGP next-hops is usually less than the number of prefixes, the lookup time is reduced.
5. After the affected BGP next-hops are identified, the RIBM and FIB issue a next-hop update to the hardware SDK to use the updated list of IGP-resolved next-hops.

6. PE3 is now reachable only through P2. Consequently, the RIBM and FIB at PE1 send a next-hop update for PE3 to the hardware SDK with the updated IGP-resolved next-hop (P2).

7. BGP next-hop PE4 continues to be reachable through P1 and P2.

BGP PIC edge node failure

The following figure illustrates a BGP PIC edge node failure and the behavior of the route tables.
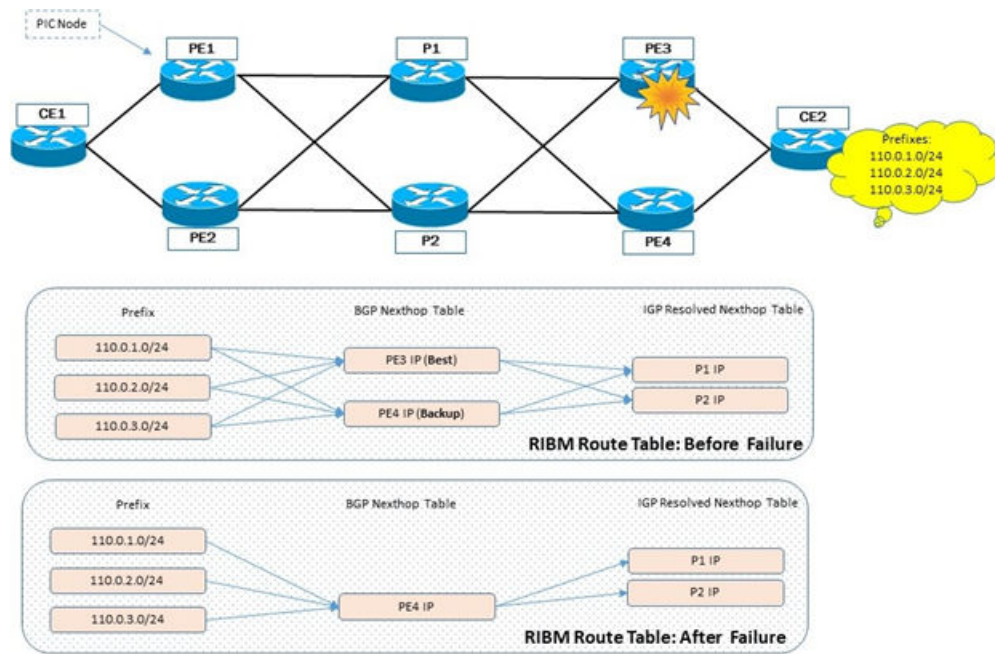
Figure 20: BGP PIC edge node failure

The behavior is as follows:

1. BGP does not form ECMP and instead provides the backup next-hop (PE4) and the primary next-hop (PE3). Only primary next-hop is programmed in the hardware SDK.

2. When PE3 fails, the RIBM and FIB at PE1 are notified by BGP that PE3 is not reachable.

3. The RIBM and FIB issue a next-hop update to the switch from PE3 to PE4 to the hardware SDK.

4. Traffic now flows through PE4.

BGP with eBGP primary-to-backup failure

The following figure illustrates a BGP with eBGP primary-to-backup failure and the behavior of the route tables.
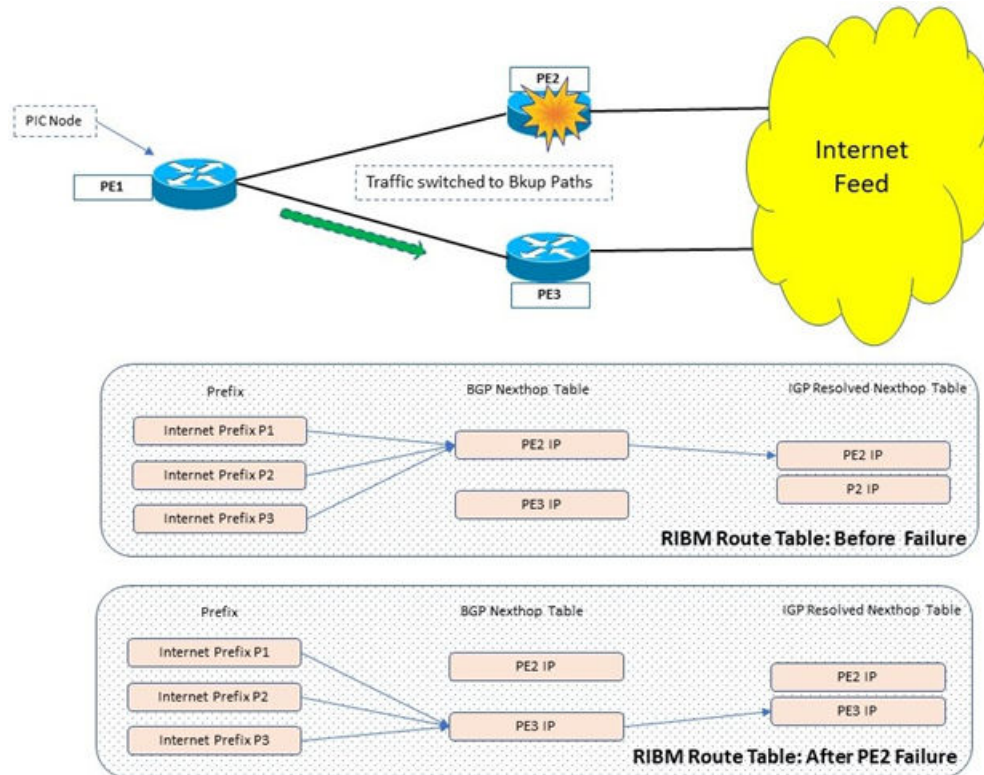
Figure 21: BGP with eBGP primary-to-backup failure

The behavior is as follows:

1. BGP does not form ECMP and instead provides backup next-hop PE3 and primary next-hop PE2. Only the primary next-hop is programmed in the hardware SDK.
2. When PE2 fails, the RIBM and FIB at PE1 are notified by BGP and conclude that PE3 is not reachable.
3. The RIBM and FIB issue a next-hop update to the switch from PE2 to PE3 to the hardware SDK.
4. Traffic now flows through PE3.

BGP with eBGP backup-to-primary failure

The following figure illustrates a BGP with eBGP backup-to-primary failure and the behavior of the route tables.
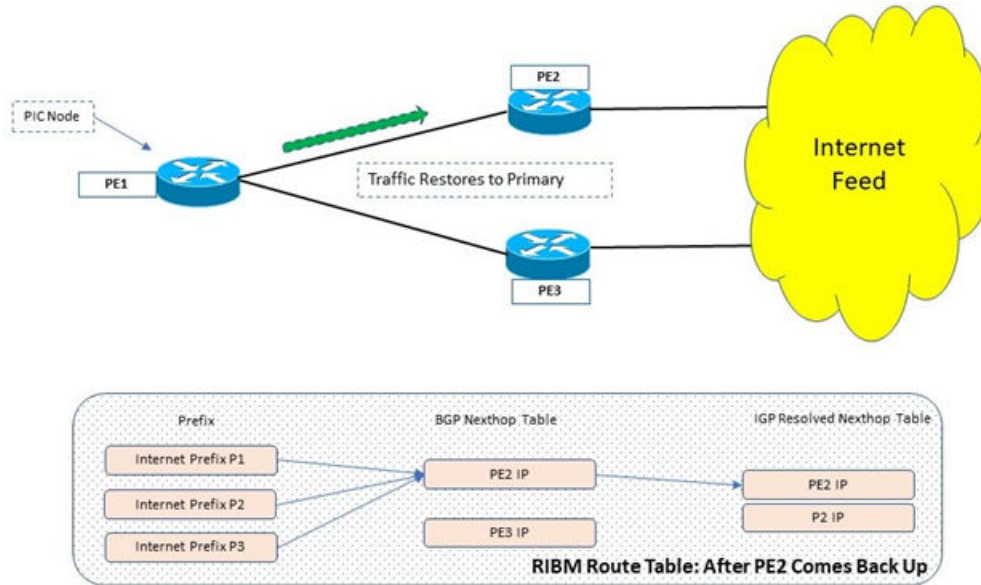
Figure 22: BGP with eBGP backup-to-primary failure

The behavior is as follows:

1. If PE2 comes back alive and gives the same set of routes to add to the RIBM, the primary path is treated as restored and traffic is switched back to the primary.

BGP PIC Considerations

Note the following considerations for this feature.

- BGP PIC is supported for IPv4 and IPv6 address families. It is not supported for Layer 3 VPN, EVPN, or IP over MPLS (IPoMPLS).
- BGP PIC is disabled by default.
- BGP PIC is applicable across all VRFs and supported address families.
- Both PIC core and PIC edge use cases are supported.
- PIC is supported on the SLX 9640 and the SLX 9540 series devices.
- PIC and high availability (HA) are supported. BGP graceful restart must be configured for PIC HA to work.
- GBP PIC is compatible with Optiscale profiles.

Enable BGP PIC

You can enable BGP Prefix-Independent Convergence to accelerate data path convergence under failover conditions.

About This Task

BGP PIC is disabled by default.

Procedure

1. Enter global configuration mode.

```
device# configure terminal
```

2. Enable BGP.

```
device(config)# prefix-independent-convergence
```

3. Confirm the configuration. Backup paths are not displayed in the output.

```
device(config)# show running-config
```

4. Confirm changes in the next-hop path.

```
device(config)# show ip route
IP Routing Table for VRF "default-vrf"
Total number of IP routes: 2
'*' denotes best ucast next-hop
'[x/y]' denotes [preference/metric]

10.20.30.0/24,
    *via DIRECT, Null0, [1/1], 0m11s, static, tag 0
10.20.31.0/24,
    *via DIRECT, Null0, [1/1], 0m5s, static, tag 0
```

Although the route remains untouched in hardware, the timestamps (highlighted in the example) reflect changes in the next-hop path.

Example

The following example summarizes the commands in this procedure.

```
device# configure terminal
device(config)# prefix-independent-convergence
device(config)# show running-config
device(config)# show ip route
```

BGP Fast Convergence with Delayed Route Calculation

BGP delayed route calculation delays the BGP BEST-path selection until BGP receives the route update information from its RIB-IN peers. This calculation minimizes the number of times that the BGP BEST-path decision process runs and improves the efficiency of the route updates (RIB-OUT) computation.

Delayed route calculation overview

By default, the BGP process accepts multiple incoming routing updates, computes the BEST-path selection, and advertises this selection to its peers immediately. When a BGP router reloads and comes up, BGP does not possess all of the route update information from its RIB-IN peers before it starts its BEST-path selection. Therefore, BGP processes incoming updates and computes the BEST-path several times. The scale of the routing updates from multiple RIB-IN peers makes BGP inefficient in calculating the BEST-path and generating RIB-OUT. Also, complex route policies of prefix lists or route-map filters for RIB-OUT peers add significant delay in generating RIB-OUT.

BGP delayed route calculation improves BGP convergence in this BGP protocol operation. A BGP peer starts propagating its route updates after it sends an initial explicit keepalive message to indicate that the session is up (ESTABLISHED). Until the peer is done propagating its route updates, most implementations do not send a second explicit keepalive message. The duration between the initial keepalive message and the second keepalive message is the duration in which the peer is propagating

its bulk route updates. Based on certain heuristics and events, the BGP process can delay its BEST-path selection. In this delay (or learning) phase, BGP does not make BEST-path decisions, does not install routes in the RIB or hardware, and does not generate RIB-OUT.

BGP peer learning phase

When a BGP router comes up after a reload, each BGP peer is placed in learning phase after receiving the first keepalive from the peer, which the peer sends when the peer session transitions from OPEN_CONFIRM to ESTABLISHED.

A peer that is in learning phase is denoted by the notation "$" concatenated to the "ESTAB" state (for example, the `show ip/ipv6 bgp summary` command displays ESTAB$).

A peer that comes up as ESTABLISHED is placed in learning phase only if the VRF and Subsequent Address Family Identifier (SAFI) instance is in read-only mode. The read-only mode timer for a VRF and SAFI instance starts when the first peer comes up in that instance, according to the delay settings.

Table 23: BEST path selection delay

Setting	Description
min-delay	The minimum time that a peer spends in read-only mode and by which the BGP-BEST path selection is delayed. The default is 180 seconds. This time allows all BGP peers in the VRF and SAFI instance to come up and participate in BGP read-only mode.
max-delay	The maximum time that a peer spends in read-only mode and by which the BGP-BEST path selection is delayed.
msg-idle-time	The number of seconds to wait for an update from a peer before moving the peer out of the learning phase. The default is 2 seconds.

The key to fast convergence is detecting the end of the learning phase for each peer as early as possible. A peer can be detected and moved out of learning phase based on the following events. When all of the peers in the VRF and SAFI instance have completed the learning phase, the route calculation is scheduled immediately for this instance.

Table 24: End-of-learning-phase events

Event	Description
Second keepalive is received	Probed when the keepalive is received for a peer.
One of the following: • No update message from the peer • Time difference between subsequent update message is greater than the message idle time	• Probed when the update message is received from the peer. • Probed in the periodic timer, every 5 seconds.
Minimum time in read-only mode	The BGP-BEST path selection is delayed at least by the minimum time. This time allows all BGP peers in the instance to participate in BGP read-only mode.

Table 24: End-of-learning-phase events (continued)

Event	Description
Maximum time in read-only mode	• A peer can continuously send route updates. Leaving such a situation forcefully ends the learning phase for a peer. • A timer starts when the first peer in the VRF and SAFI instance is placed in the learning phase. When the timer starts, 300 seconds are allowed (by default) for all peers to complete their learning phase. • Probed in a periodic timer every 5 seconds. Peers that are still in the learning phase after 300 seconds of the learning phase start time are forcefully moved out of learning phase.
A peer flaps, is removed through configuration, or is shutdown (admin disable)	The learning phase for the peer is forcefully ended.

At each probing point (processing keepalive, processing updates, periodic timer, and peer session state change), if it is detected that all BGP peers in the VRF and SAFI instance have completed their learning phase, BGP BEST-PATH selection is immediately scheduled for the instance.

Configure BGP Delayed Route Calculation

BGP delayed route calculation delays the BGP BEST-path selection until BGP receives the route update information from its RIB-IN peers.

About This Task

BGP delayed route calculation is disabled by default. When you enable the calculation, the default minimum delay, maximum delay, and message idle time are also enabled. You can change these values. For more information, see BGP Fast Convergence with Delayed Route Calculation on page 218.

Table 25: BEST path selection delay

Setting	Description
min-delay	The minimum time that a peer spends in read-only mode and by which the BGP-BEST path selection is delayed. The default is 180 seconds.
max-delay	The maximum time that a peer spends in read-only mode and by which the BGP-BEST path selection is delayed.
msg-idle-time	The number of seconds to wait for an update from a peer before moving the peer out of the learning phase. The default is 2 seconds.

Procedure

1. Access global configuration mode.
   ```
   device# configure terminal
   ```
2. Access BGP router configuration mode.
   ```
   device(config)# router bgp
   ```

3. Enable BGP delayed route calculation.

```
device(config-bgp-router)# init-route-calc-delay
```

4. (Optional) Configure the minimum delay in seconds.

```
device(config-bgp-router)# init-route-calc-delay min-delay 360
```

This example configures a minimum delay of 360 seconds.

5. (Optional) Configure the maximum delay in seconds.

```
device(config-bgp-router)# init-route-calc-delay max-delay 600
```

This example configures a maximum delay of 600 seconds.

6. (Optional) Configure the message idle time in seconds.

```
device(config-bgp-router)# init-route-calc-delay msg-idle-time 10
```

This example configures a message idle time of 10 seconds.

Example

The following example summarizes the commands in this procedure.

```
device# configure terminal
device(config)# router bgp
device(config-bgp-router)# init-route-calc-delay
device(config-bgp-router)# init-route-calc-delay min-delay 360
device(config-bgp-router)# init-route-calc-delay max-delay 600
device(config-bgp-router)# init-route-calc-delay msg-idle-time 10
```

BGP4+

BGP4+ Overview

The SLX-OS device supports IPv6 multiprotocol BGP (MBGP) extensions that allow Border Gateway Protocol version 4 plus (BGP4+) to distribute routing information.

BGP4+ supports the same features and functionality as IPv4 BGP (BGP4). For more information, see BGP4 Overview on page 124.

IPv6 MBGP enhancements include:

- An IPv6 unicast address family and network layer reachability information (NLRI)
- Next hop attributes that use IPv6 addresses

> **Note**
> BGP4+ supports the advertising of BGP4+ unicast routes among different address families. It does not 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
  auto-shutdown-new-neighbors  Auto shutdown new neighbor
  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
  describe                  Display transparent command information
  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 asleast
                            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
```

```
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.

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.

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
  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
  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
```

Configure a BGP4+ Neighbor with a Global IPv6 Address

You can configure a BGP4+ neighbor with a global IPv6 address.

Procedure

1. Access global configuration mode.

```
device# configure terminal
```

2. Access BGP configuration mode.

```
device(config)# router bgp
```

3. Specify the autonomous system in which your device resides.

```
device(config-bgp-router)# local-as 1000
```

4. Specify the autonomous system in which the remote neighbor resides.

```
device(config-bgp-router)# neighbor 2001:db8:93e8:cc00::1 remote-as 1001
```

This example specifies the **neighbor** command, the neighbor address of 2001:db8:93e8:cc00::1, the **remote-as** keyword, and autonomous system 1001.

5. Access IPv6 address family configuration mode.

```
device(config-bgp-router)# address-family ipv6 unicast
```

6. Enable the exchange of information with the neighbor.

```
device(config-bgp-ipv6u)# neighbor 2001:db8:93e8:cc00::1 activate
```

This example specifies the **neighbor** command, the neighbor address, and the **activate** keyword.

Example

The following example summarizes the commands in this procedure.

```
device# configure terminal
device(config)# 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
```

Configure a BGP4+ Neighbor with a Link-local Address

You can configure a BGP4+ neighbor with a link-local address.

Procedure

1. Access global configuration mode.

   device# configure terminal

2. Access BGP configuration mode.

```
device(config)# router bgp
```

3. Specify the autonomous system which your device resides.

```
device(config-bgp-router)# local-as 1000
```

4. Specify the autonomous system in which the remote neighbor resides.

```
device(config-bgp-router)# neighbor fe80:4398:ab30:45de::1 remote-as 1001
```

This example specifies the **neighbor** command, the neighbor address of fe80:4398:ab30:45de::1, the **remote-as** keyword, and autonomous system 1001.

5. Specify the interface through which the neighbor and the local device communicate to exchange prefixes.

```
device(config-bgp-router)# neighbor fe80:4398:ab30:45de::1 update-source ethernet 3/1
```

This example specified the **neighbor** command, the neighbor address, the **update-source** keyword, and Ethernet interface slot 3, port 1.

6. Access IPv6 address family configuration mode.

```
device(config-bgp-router)# address-family ipv6 unicast
```

7. Enable the exchange of information with the neighbor.

```
device(config-bgp-ipv6u)# neighbor fe80:4398:ab30:45de::1 activate
```

This example specifies the **neighbor** command, the neighbor address, and the **activate** keyword.

8. Apply a route map to outgoing routes.

```
device(config-bgp-ipv6u)# neighbor fe80:4398:ab30:45de::1 route-map out myroutemap
```

This example specifies the **neighbor** command, the neighbor address, the **route-map out** keyword, and the route map titled myroutemap.

9. Return to global configuration mode.

```
device(config-bgp-ipv6u)# exit
device(config-bgp-router)# exit
device(config)#
```

10. Define the route map and enter route map configuration mode.

```
device(config)# route-map myroutemap permit 10
```

This example specifies the **route-map** command, the name of the route map, and a matching pattern. The route map sets up the global next hop for packets that are destined for the neighbor.

11. Specify the IPv6 address of the next hop.

```
device(config-routemapmap-myroutemap/permit/10)# set ipv6 next-hop 2001::10
```

Example

The following example summarizes the commands in this procedure.

```
device# configure terminal
device(config)# 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 ethernet 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)# route-map myroutemap permit 10
device(config-routemapmap-myroutemap/permit/10)# set ipv6 next-hop 2001::10
```

BGP4+ Peer Groups

You can use the **neighbor peer-group** command to group IPv6 neighbors that have the same attributes and parameters.

You can associate neighbor IPv6 addresses with a BGP4+ peer group. All of the attributes that are allowed on a neighbor are also allowed on a peer group.

An attribute value configured for a neighbor takes precedence over the attribute value configured for a peer group. The default attribute is used if the neither the peer group nor the neighbor has the attribute configured.

BGP4+ peers and peer groups are activated in IPv6 address family configuration mode.

Create a BGP4+ Peer Group

A BGP4+ peer group is a collection of IPv6 neighbors that have the same attributes and parameters

Procedure

1. Access global configuration mode.

   ```
   device# configure terminal
   ```

2. Access BGP configuration mode.

   ```
   device(config)# router bgp
   ```

3. Create a peer group.

   ```
   device(config-bgp-router)# neighbor mypeergroup1 peer-group
   ```

   This example shows the **neighbor** command, the peer group name (mypeergroup1), and the **peer-group** keyword.

4. Specify the autonomous system number (ASN) of the peer group.

   ```
   device(config-bgp-router)# neighbor mypeergroup1 remote-as 11
   ```

   This example associates ASN 11 to the new peer group.

5. Associate a neighbor with the peer group.

   ```
   device(config-bgp-router)# neighbor 2001:2018:8192::125 peer-group mypeergroup1
   ```

   This example shows the **neighbor** command, the IPv6 address of the neighbor, the **peer-group** keyword, and the name of the peer group (mypeergroup1).

6. Repeat step 5 as needed to associate more neighbors with the peer group.

7. Access IPv6 address family configuration mode.

   ```
   device(config-bgp-router)# address-family ipv6 unicast
   ```

8. Activate an IPv6 BGP session with the new peer group.

   ```
   device(config-bgp-ipv6u)# neighbor mypeergroup1 activate
   ```

Example

The following example summarizes the commands in this procedure

```
device# configure terminal
device(config)# router bgp
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)# address-family ipv6 unicast
device(config-bgp-ipv6u)# neighbor mypeergroup1 activate
```

Create a BGP Peer Group with IPv4 and IPv6 Peers

A BGP peer group can contain both IPv4 and IPv6 peers.

Procedure

1. Access global configuration mode.

    ```
    device# configure terminal
    ```

2. Access BGP configuration mode.

    ```
    device(config)# router bgp
    ```

3. Create a peer group.

    ```
    device(config-bgp-router)# neighbor p1 peer-group
    ```

    This example shows the **neighbor** command, the peer group name (p1), and the **peer-group** keyword.

4. Specify the autonomous system number (ASN) of the peer group.

    ```
    device(config-bgp-router)# neighbor p1 remote-as 11
    ```

    This example associates ASN 11 to the new peer group.

5. Associate an IPv6 neighbor with the peer group.

    ```
    device(config-bgp-router)# neighbor 2001:2018:8192::124 peer-group p1
    ```

    This example shows the **neighbor** command, the IPv6 address of the neighbor, the **peer-group** keyword, and the name of the peer group (p1).

6. Associate an IPv4 neighbor with the peer group.

    ```
    device(config-bgp-router)# neighbor 10.0.0.1 peer-group p1
    ```

    This example shows the **neighbor** command, the IPv4 address of the neighbor, the **peer-group** keyword, and the name of the peer group (p1).

7. Repeat steps 5 and 6 as needed to associate more neighbors with the peer group.

8. Access IPv6 address family configuration mode.

    ```
    device(config-bgp-router)# address-family ipv6 unicast
    ```

9. Activate an IPv6 BGP session with the new peer group.

    ```
    device(config-bgp-ipv6u)# neighbor p1 activate
    ```

Example

The following example summarizes the commands in this procedure.

```
device# configure terminal
device(config)# router bgp
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
```

Enable ASN Capability Globally for BGP4

You can configure 4-byte autonomous system number (ASN) capability at the BGP4 global level.

Procedure

1. Access global configuration mode.

   ```
   device# configure terminal
   ```

2. Access BGP configuration mode.

   ```
   device(config)# router bgp
   ```

3. Enable 4-byte ASN capability.

   ```
   device(config-bgp-router)# capability as4-enable
   ```

Enabling ASN capability for a BGP4+ neighbor

Four-byte autonomous system numbers (ASNs) can be configured for a neighbor or peer-group. The following task enables 4-byte autonomous system number (ASN) capability for a BGP4+ neighbor.

About This Task

Procedure

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

   ```
   device# configure terminal
   ```

2. Enter the **router bgp** command to enable BGP routing.

   ```
   device(config)# router bgp
   ```

3. Enter the **local-as** command to configure the autonomous system number (ASN) in which your device resides.

   ```
   device(config-bgp-router)# local-as 1000
   ```

4. Enter the **neighbor capability as4** command with the **enable** parameter, and specify an IPv6 address, to enable 4-byte autonomous system number (ASN) capability for a specific neighbor.

   ```
   device(config-bgp-router)# neighbor 2001:db8:93e8:cc00::1 capability as4 enable
   ```

Example

The following example enables 4-byte autonomous system number (ASN) capability for a specific BGP4+ neighbor.

```
device# configure terminal
device(config)# router bgp
device(config-bgp-router)# local-as 1000
device(config-bgp-router)# neighbor 2001:db8:93e8:cc00::1 capability as4 enable
```

Importing routes into BGP4+

Routes can be explicitly specified for advertisement by BGP.

Before You Begin

The routes imported into BGP4+ must first exist in the IPv6 unicast route table.

Procedure

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

   ```
   device# configure terminal
   ```

2. Enter the **router bgp** command to enable BGP routing.

   ```
   device(config)# router bgp
   ```

3. 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
   ```

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 **network** command and specify a *network/mask* to import the specified prefix into the BGP4+ database.

   ```
   device(config-bgp-ipv6u)# network 2001:db8::/32
   ```

Example

The following example imports the 2001:db8::/32 prefix in to the BGP4+ database for advertising.

```
device# configure terminal
device(config)# 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.

Before You Begin

The default route must be present in the local IPv6 route table.

Procedure

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

   ```
   device# configure terminal
   ```

2. Enter the **router bgp** command to enable BGP routing.

```
device(config)# router bgp
```

3. Enter the **address-family ipv6 unicast** command to enter IPv6 address family unicast configuration mode.

```
device(config-bgp-router)# address-family ipv6 unicast
```

4. 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
```

Example

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)# 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.

About This Task

Procedure

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

```
device# configure terminal
```

2. Enter the **router bgp** command to enable BGP routing.

```
device(config)# router bgp
```

3. Enter the **local-as** command to configure the autonomous system number (ASN) in which your device resides.

```
device(config-bgp-router)# local-as 1000
```

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 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
```

Example

The following example enables a BGP4+ device to advertise the default IPv6 route to a specific neighbor.

```
device# configure terminal
device(config)# 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
```

Redistribute Routes into BGP4+

You can redistribute static, connected, OSPF, and IS-IS routes into the BGP4+ routing table.

About This Task
The examples in this procedure redistribute OSPFv3 routes into BGP4+. For other examples, see the *Extreme SLX-OS Command Reference*.

Procedure

1. Access global configuration mode.

   ```
   device# configure terminal
   ```

2. Access BGP configuration mode.

   ```
   device(config)# router bgp
   ```

3. Access IPv6 address family configuration mode.

   ```
   device(config-bgp-router)# address-family ipv6 unicast
   ```

4. Enter the **redistribute** command using

   ```
   device(config-bgp-ipv6u)# redistribute ospf match external1
   ```

   This example uses the **ospf** and **external1** keywords to redistribute IPv6 OSPF external type 1 routes.

Example

The following example summarizes the commands in this procedure.

```
device# configure terminal
device(config)# router bgp
device(config-bgp-router)# address-family ipv6 unicast
device(config-bgp-ipv6u)# redistribute ospf match external1
```

Configure the IPv6 Default Route as a Valid BGP4+ Next Hop

By default, a device does not use a default route to resolve a BGP4+ next-hop route.

About This Task
If the IPv6 route lookup for the BGP4+ next-hop does not result in a valid Interior Gateway Protocol (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.

Procedure

1. Access global configuration mode.

```
device# configure terminal
```

2. Access BGP configuration mode.

```
device(config)# router bgp
```

3. Access IPv6 address family configuration mode.

```
device(config-bgp-router)# address-family ipv6 unicast
```

4. Configure the device to use the default route as a valid next hop.

```
device(config-bgp-ipv6u)# next-hop-enable-default
```

Example

The following example summarizes the commands in this procedure.

```
device# configure terminal
device(config)# router bgp
device(config-bgp-router)# address-family ipv6 unicast
device(config-bgp-ipv6u)# next-hop-enable-default
```

BGP4+ Recursive Next-Hop Lookups

With recursive next-hop lookups, 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 if 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 situation 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 does not contain a route that can be used to reach the BGP4+ route destination.

You can use recursive next-hop lookups to enable the device to find the IGP route to the next-hop gateway for a BGP4+ route. With this feature enabled, if the first lookup for a BGP4+ route results in an IBGP path that originated in the same AS, rather than in an IGP path or static route path, the device performs a lookup on the next-hop IPv6 address for the next-hop gateway.

If the 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.

> **Note**
> You must configure a static route or use an IGP to learn the route to the EBGP multihop peer.

Enable BGP4+ Recursive Next-hop Lookups

With recursive next-hop lookups, a device can find the IGP route to the next-hop gateway for a BGP4+ route.

Procedure

1. Access global configuration mode.

```
device# configure terminal
```

2. Access BGP configuration mode.

```
device(config)# router bgp
```

3. Access IPv6 address family configuration mode.

```
device(config-bgp-router)# address-family ipv6 unicast
```

4. Enable recursive next-hop lookups.

```
device(config-bgp-ipv6u)# next-hop-recursion
```

Example

The following example summarizes the commands in this procedure.

```
device# configure terminal
device(config)# router bgp
device(config-bgp-router)# address-family ipv6 unicast
device(config-bgp-ipv6u)# next-hop-recursion
```

BGP4+ Multiprotocol Extensions for NLRI

Multiprotocol BGP (MBGP) is an extension to BGP4+ that enables BGP to carry routing information for multiple address families.

The following BGP4+ attributes handle multiprotocol extensions for BGP:

* Multiprotocol reachable Network Layer Reachability Information (MP_REACH_NLRI): Used to carry the set of reachable destinations and next-hop information, to use 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. A BGP4+ speaker that does not support the multiprotocol capabilities ignores the information 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 in the MP_REACH_NLRI path attribute defines the network layer address of the border router that is 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 use the **neighbor route-reflector-client** command to configure a BGP peer as a route-reflector client from the device that is going to reflect the routes (the route reflector).

Multiple route reflectors should belong to the same cluster. You can use the **cluster-id** command to change the device ID.

The route-reflector client reflects the routes as follows:

- Routes from the client are reflected to the client and to non-client peers.
- Routes from non-client peers are reflected only to client peers.

If route-reflector clients are connected in a full IBGP mesh, you can use the **no client-to-client-reflection** command to disable client-to-client reflection on the route reflector

A BGP device advertises only preferred routes that are installed in the Routing Table Manager (RTM). When a route cannot be installed in the RTM because the routing table is full, the route reflector may not reflect that route. You can use the **always-propagate** command to configure the route reflector to reflect routes that are not in the RTM.

Configure a Cluster ID for a BGP4+ Route Reflector

When you have multiple route reflectors, change the cluster ID so that all route reflectors belong to the same cluster.

Procedure

1. Access global configuration mode.
   ```
   device# configure terminal
   ```
2. Access BGP configuration mode.
   ```
   device(config)# router bgp
   ```
3. Specify the autonomous system in which your device resides.
   ```
   device(config-bgp-router)# local-as 1000
   ```
4. Change the cluster ID of a device from the default device ID.
   ```
   device(config-bgp-router)# cluster-id 321
   ```

Example

The following example summarizes the commands in this procedure.

```
device# configure terminal
device(config)# router bgp
```

```
device(config-bgp-router)# local-as 1000
device(config-bgp-router)# cluster-id 321
```

Configure a BGP4+ Route-Reflector Client

You configure a BGP peer as a route-reflector client from the device that is the route reflector.

Procedure

1. Access global configuration mode.

   ```
   device# configure terminal
   ```

2. Access BGP configuration mode.

   ```
   device(config)# router bgp
   ```

3. Specify the autonomous system in which your device resides.

   ```
   device(config-bgp-router)# local-as 1000
   ```

4. Access IPv6 address family configuration mode.

   ```
   device(config-bgp-router)# address-family ipv6 unicast
   ```

5. Specify the neighbor to be the route-reflector client.

   ```
   device(config-bgp-ipv6u)# neighbor 2001:db8:e0ff:783a::4 route-reflector-client
   ```

Example

The following example summarizes the commands in this procedure.

```
device# configure terminal
device(config)# 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
```

Aggregate the Routes to Advertise to BGP4+ Neighbors

By default, a device advertises individual routes for all networks. To minimize the size of the routing table, you can aggregate the routes in a range of networks into one IPv6 prefix.

About This Task

You can aggregate BGP4+ routes into one network prefix, for example, 2001:db8::/24. This single prefix is advertised to BGP4+ neighbors instead of the individual BGP routes in the routing table that are within the specified range.

Procedure

1. Access global configuration mode.

   ```
   device# configure terminal
   ```

2. Access BGP configuration mode.

   ```
   device(config)# router bgp
   ```

3. Access IPv6 address family configuration mode.

   ```
   device(config-bgp-router)# address-family ipv6 unicast
   ```

4. Specify the network prefix into which routes are to be aggregated.

```
device(config-bgp-ipv6u)# aggregate-address 2001:db8::/32
```

Example

The following example summarizes the commands in this procedure.

```
device# configure terminal
device(config)# router bgp
device(config-bgp-router)# address-family ipv6 unicast
device(config-bgp-ipv6u)# aggregate-address 2001:db8::/32
```

BGP4+ Multipath for Load Balancing

By default, BGP4+ balances traffic across one path. You can use the multipath feature to enable load-balancing across different BGP4+ paths.

BGP4+ selects only one best path for each IPv6 prefix it receives before installing it in the IP routing table. For load balancing across different paths, use the **maximum-paths** command to enable BGP4+ multipath.

You can select iGP paths, eBGP paths, or a combination of both.

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

Enable Load Balancing Across Different BGP4+ Paths

You can use the BGP4+ multipath feature to enable load balancing different BGP4+ paths.

About This Task
You can use the num variable to specify a maximum number of shared paths. Or you can use the **use-load-sharing** keyword to use the value that has already configured by means of the **ip load-sharing** command.

Procedure

1. Access global configuration mode.
```
device# configure terminal
```

2. Access BGP configuration mode.
```
device(config)# router bgp
```

3. Access IPv6 address family configuration mode.
```
device(config-bgp-router)# address-family ipv6 unicast
```

4. Perform one of the following steps to specify load-sharing paths:

- Use the *num* variable to specify a maximum number of shared paths.

  ```
  device(config-bgp-ipv6u)# maximum-paths 8
  ```

- Use the value that has been configured by means of the **ip load-sharing** command.

  ```
  device(config-bgp-ipv6u)# maximum-paths use-load-sharing
  ```

Example

The following example summarizes the commands in this procedure

```
device# configure terminal
device(config)# router bgp
device(config-bgp-router)# address-family ipv6 unicast
device(config-bgp-ipv6u)# maximum-paths 8
device(config-bgp-ipv6u)# maximum-paths use-load-sharing
```

BGP4+ 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.

By default, route maps that are applied in IPv4 address family configuration mode with the **neighbor route-map** command apply only to IPv4 unicast address prefixes. To apply route maps to IPv6 unicast address prefixes, use the same command in IPv6 address family configuration mode.

The route maps are applied as the inbound or outbound routing policy for neighbors under the address family. Configuring separate route maps under each address family type simplifies managing complicated or different policies for each address family.

Configure a Route Map for BGP4+ Prefixes

You can apply route maps to IPv6 unicast address prefixes either as the inbound or outbound routing policy for neighbors in IPv6 address family configuration mode.

Procedure

1. Access global configuration mode.

   ```
   device# configure terminal
   ```

2. Configure an IPv6 prefix list to filter traffic.

   A prefix list consists of at least one conditional statement that executes an action if a route matches a specified prefix.

   ```
   device(config)# ipv6 prefix-list myprefixlist permit 2001:db8::/32
   ```

   This example creates a prefix list named myprefixlist that allows routes with the prefix of 2001:db8::/32.

3. Create a route map instance and allow a matching pattern.

```
device(config)# route-map myroutemap permit 10
```

This example creates a route map called myroutemap and allows a matching pattern of 10.

4. Configure the route map to match IPv6 routes that are specified by the prefix list.

```
device(config-route-map-myroutemap/permit/10)# match ipv6 address prefix-list
myprefixlist
```

5. Return to global configuration mode.

```
device(config-route-map-myroutemap/permit/10)# exit
```

6. Access BGP configuration mode.

```
device(config)# router bgp
```

7. Specify the autonomous system in which your device resides.

```
device(config-bgp-router)# local-as 1000
```

8. Specify the autonomous system in which the remote neighbor resides.

```
device(config-bgp-router)# neighbor fe80:4398:ab30:45de::1 remote-as 1001
```

9. Access IPv6 address family configuration mode.

```
device(config-bgp-router)# address-family ipv6 unicast
```

10. Enable the exchange of information with the neighbor.

```
device(config-bgp-ipv6u)# neighbor fe80:4398:ab30:45de::1 activate
```

11. Apply the route map to the neighbor's outgoing routes.

```
device(config-bgp-ipv6u)# neighbor fe80:4398:ab30:45de::1 route-map out myroutemap
```

For an example of applying a route map as the inbound routing policy, see the *Extreme SLX-OS Command Reference*.

Example

This example summarizes the commands in this procedure.

```
device# configure terminal
device(config)# ipv6 prefix-list myprefixlist seq 10 permit 2001:db8::/32
device(config)# 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)# 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
```

Change the Weight Added to Receive BGP4+ Routes

The device adds weight to routes that are received from BGP neighbors. In the BGP4+ path-selection algorithm, routes with larger weights have preference over routes with smaller weights.

About This Task
You can change the weight of routes that are received from a specified BGP4+ neighbor.

Procedure

1. Access global configuration mode.

   ```
   device# configure terminal
   ```

2. Access BGP configuration mode.

   ```
   device(config)# router bgp
   ```

3. Specify autonomous system in which your device resides.

   ```
   device(config-bgp-router)# local-as 65520
   ```

4. Access address family IPv6 unicast configuration mode.

   ```
   device(config-bgp-router)# address-family ipv6 unicast
   ```

5. Specify the neighbor and the weight that you want to add to routes that are received from that neighbor.

   ```
   device(config-bgp-ipv6u)# neighbor 2001:db8:e0ff:783a::4 weight 200
   ```

   This examples shows the **neighbor** command, the IPv6 address of the neighbor, the **weight** keyword, and the weight value of 200.

Example

The following example summarizes the commands in this procedure.

```
device# configure terminal
device(config)# 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
```

Enable 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. From there, you can access several commands that allow the configuration of BGP4+ for the non-default VRF instance.

Before You Begin
Before performing this task, ensure that a non-default VRF instance is configured.

Procedure

1. Access global configuration mode.

   ```
   device# configure terminal
   ```

2. Access BGP configuration mode.

   ```
   device(config)# router bgp
   ```

3. Specify the VRF name and enter BGP address-family IPv6 unicast VRF configuration mode.

   ```
   device(config-bgp-router)# address-family ipv6 unicast vrf green
   ```

   This example shows the **address-family ipv6 unicast** command, the **vrf** keyword, and the VRF name (green).

Example

The following example summarizes the commands in this procedure.

```
device# configure terminal
device(config)# router bgp
device(config-bgp-router)# address-family ipv6 unicast vrf green
device(config-bgp-ipv6u-vrf)#
```

BGP4+ outbound route filtering

You use the BGP4+ Outbound Route Filtering (ORF) feature to minimize the number of BGP updates sent between BGP peers.

When ORF is enabled, unwanted routing updates are removed, 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. The remote peer applies the filter as an outbound filter for the neighbor.

You can configure ORF 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. Using send mode, the local peer exchanges the ORF capability with a remote peer for a prefix list that is configured as an inbound filter for that peer locally.

The remote peer sends the first update after it receives a ROUTE REFRESH request or BGP ORF with IMMEDIATE from the peer. The local and remote peers exchange updates to maintain the ORF on each router.

Configure BGP4+ Outbound Route Filtering

You can configure BGP4+ Outbound Route Filtering (ORF) in receive mode, send mode, or both, minimizing the number of BGP updates exchanged between BGP peers.

Procedure

1. Access global configuration mode.

   ```
   device# configure terminal
   ```

2. Access BGP configuration mode.

   ```
   device(config)# router bgp
   ```

3. Access IPv6 address family configuration mode.

   ```
   device(config-bgp-router)# address-family ipv6 unicast
   ```

4. (For send mode only) Filter the incoming route updates from a specified BGP neighbor.

   ```
   device(config-bgp-ipv6u)# neighbor 2001:db8:e0ff:783a::4 prefix-list myprefixlist in
   ```

   This example specifies the **neighbor prefix-list** command, the IPv6 address of the neighbor, a prefix list named myprefixlist, and the keyword **in**.

5. To advertise ORF send capabilities, run the **neighbor capability orf prefixlist** command with the **send** keyword.

   ```
   device(config-bgp-ipv6u)# neighbor 2001:db8:e0ff:783a::4 capability orf prefixlist send
   ```

6. To advertise ORF receive capabilities, run the **neighbor capability orf prefixlist** command with the **receive** keyword.

```
device(config-bgp-ipv6u)# neighbor 2001:db8:e0ff:783a::4 capability orf prefixlist
receive
```

7. To advertise both ORF send and receive capabilities, run the **neighbor capability orf prefixlist** command.

```
device(config-bgp-ipv6u)# neighbor 2001:db8:e0ff:783a::4 capability orf prefixlist
```

Example

The following example summarizes the commands for advertising ORF in receive mode.

```
device# configure terminal
device(config)# router bgp
device(config-bgp-router)# address-family ipv6 unicast
device(config-bgp-ipv6u)# neighbor 2001:db8:e0ff:783a::4 capability orf prefixlist receive
```

Example

The following example summarizes the commands for advertising ORF in send mode.

```
device# configure terminal
device(config)# router bgp
device(config-bgp-router)# address-family ipv6 unicast
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
```

Example

The following example summarizes the commands for advertising ORF in both send and receive modes.

```
device# configure terminal
device(config)# router bgp
device(config-bgp-router)# address-family ipv6 unicast
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 sub-autonomous (sub-AS) systems and grouped into one BGP4 confederation.

Each sub-AS system must be uniquely identified in the confederation AS by a sub-AS system number. In each sub-AS system, all the rules of internal BGP (iBGP) apply. For example, all BGP routers in the sub-AS system must be fully meshed. Although eBGP is used between sub-AS systems, the sub-AS systems in the confederation exchange routing information like iBGP peers. Next hop, Multi Exit Discriminator (MED), and local preference information is preserved when it crosses sub-AS system boundaries. To the outside world, a confederation looks like one AS.

The AS path list is a loop-avoidance mechanism used to detect routing updates that leave one sub-AS system and attempt to re-enter the same sub-AS system. A routing update that attempts to re-enter the sub-AS system from which it originated is detected because the sub-AS system sees its own system number listed in the update's AS path.

Configure a BGP4+ Confederation

A BGP4 confederation consists of multiple sub-autonomous (sub-AS) systems.

Procedure

1. Access global configuration mode.

```
device# configure terminal
```

2. Access BGP configuration mode.

```
device(config)# router bgp
```

3. Identify the sub-AS in which your device resides.

```
device(config-bgp-router)# local-as 65520
```

4. Assign an ASN as a BGP confederation identifier.

```
device(config-bgp-router)# confederation identifier 100
```

In this example, sub-AS 65520 is now part of a confederation that belongs to autonomous system 100.

5. Specify the ASNs of all BGP peers that you want to add to the confederation.

```
device(config-bgp-router)# confederation peers 65520 65521 65522
```

Example

The following example summarizes the commands in this procedure.

```
device# configure terminal
device(config)# 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.

Define a BGP4+ Extended Community

You must define a BGP4 extended community filter before you can apply the filter.

Procedure

1. Access global configuration mode.

   ```
   device# configure terminal
   ```

2. Specify an extended community list instance.

   ```
   device(config)# ip extcommunity-list standard mylist permit soo 123:2
   ```

   This example specifies the extended community list titled mylist and allows site of origin (SOO) from network 2 in autonomous system 123.

   > **Note**
   > The **ip-extcommunity-list** command supports a range of extended instances from 100 through 500, beyond the standard range of 1 through 99.

3. Create an inbound route map instance and enter route-map configuration mode.

   ```
   device(config)# route-map extComRmap permit 10
   ```

   This example creates the extComRmap route map that allows a matching pattern, with an instance ID of 10.

4. Configure the route map to match the extended community list.

   ```
   device(config-route-map-extComRmap/permit/10)# match extcommunity mylist
   ```

5. Exit route-map configuration mode.

   ```
   device(config-route-map-extComRmap/permit/10)# exit
   ```

6. Create an outbound route map instance and enter route-map configuration mode.

   ```
   device(config)# route-map sendExtComRmap permit 10
   ```

   This example creates the sendExtComRmap route map that allows a matching pattern, with an instance ID of 10.

7. Specify the route target (RT) extended community attribute.

   ```
   device(config-route-map-sendExtComRmap/permit/10)# set extcommunity rt 3:3
   ```

   This example specifies the **set extcommunity** command, the **rt** keyword, and the *extcommunityvalue* variable.

8. Set the extended community attribute in the route-map instance.

   ```
   device(config-route-map-sendExtComRmap/permit/10)# set extcommunity soo 2:2
   ```

   This example specifies the SOO extended community attribute with the **soo** keyword and the *extcommunityvalue* variable.

Example

The following example summarizes the commands in this procedure.

```
device# configure terminal
device(config)# ip extcommunity-list standard mylist permit soo 123:2
device(config)# route-map extComRmap permit 10
device(config-route-map-extComRmap/permit/10)# match extcommunity mylist
device(config-route-map-extComRmap/permit/10)# exit
device(config)# 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
```

Apply a BGP4+ Extended Community Filter

You can apply an extended community filter to a neighbor through a route map.

Before You Begin

Define a BGP4+ Extended Community on page 245.

Procedure

1. Access global configuration mode.

   ```
   device# configure terminal
   ```

2. Define the matching conditions for the extended community.

   ```
   device(config)# ip extcommunity-list standard mylist permit rt 123:2
   ```

   This example specifies that community list 1 permits routes from network 2 in autonomous system 123.

3. Access BGP configuration mode.

   ```
   device(config)# router bgp
   ```

4. Specify the autonomous system in which your device resides.

   ```
   device(config-bgp-router)# local-as 1000
   ```

5. Specify the autonomous system in which the remote neighbor resides.

   ```
   device(config-bgp-router)# neighbor fe80:4398:ab30:45de::1 remote-as 1001
   ```

6. Access IPv6 address family configuration mode.

   ```
   device(config-bgp-router)# address-family ipv6 unicast
   ```

7. Enable the exchange of information with the neighbor.

   ```
   device(config-bgp-ipv6u)# neighbor fe80:4398:ab30:45de::1 activate
   ```

8. Apply a route map to incoming routes.

   ```
   device(config-bgp-ipv6u)# neighbor fe80:4398:ab30:45de::1 route-map in extComRmapt
   ```

   This example uses the **in** keyword to apply route map extComRmapt.

9. Apply a route map to outgoing routes.

   ```
   device(config-bgp-ipv6u)# neighbor fe80:4398:ab30:45de::1 route-map out sendExtComRmap
   ```

   This example uses the keyword **out** to apply route map sendExtComrmap.

10. 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
   ```

   This example uses the **send-community both** keywords.

Example

The following example summarizes the commands in this procedure.

```
device# configure terminal
device(config)# ip extcommunity-list standard mylist permit rt 123:2
device(config)# 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.

About This Task

> **Note**
> High availability (HA) requires GR to be enabled.

Procedure

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

   ```
   device# configure terminal
   ```

2. Enter the **router bgp** command to enable BGP routing.

   ```
   device(config)# router bgp
   ```

3. Enter the **local-as** command to configure the autonomous system number (ASN) in which your device resides.

   ```
   device(config-bgp-router)# local-as 1000
   ```

4. 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
```

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 activate** command to add a neighbor.

```
device(config-bgp-ipv6u)# neighbor 1000::1 activate
```

7. Enter the **graceful-restart** command to enable the graceful restart feature.

```
device(config-bgp-ipv6u)# graceful-restart
```

8. 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
   ```

Example

The following example enables the graceful restart feature.

```
device# configure terminal
device(config)# 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
```

Example

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)# 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
```

Example

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)# 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
```

Example

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)# 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
```

What to Do Next

Use the **clear ipv6 bgp neighbor** command with the **all** parameter for the changes to the
graceful restart parameters to take effect immediately.

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.

Configure Auto Shutdown of BGP Neighbors on Initial Configuration

Follow this procedure to enable auto shutdown of BGP neighbors on initial configuration.

Procedure

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

   ```
   device# configure terminal
   ```

2. Enter the **router bgp** command to enable BGP routing.

   ```
   device(config)# router bgp
   ```

3. Enter the **local-as** command to configure the autonomous system number (ASN) in which your device resides.

   ```
   device(config-bgp-router)# local-as 65520
   ```

4. 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
   ```

Example

This example summarizes the commands in this procedure.

```
device# configure terminal
device(config)# 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.

Before You Begin

Procedure

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

   ```
   device# configure terminal
   ```

2. Enter the **router bgp** command to enable BGP routing.

   ```
   device(config)# router bgp
   ```

3. Enter the **local-as** command to configure the autonomous system number (ASN) in which your device resides.

   ```
   device(config-bgp-router)# local-as 65520
   ```

4. 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
```

Example

The following example disables the peer shutdown state and begins the BGP4+ session establishment process.

```
device# configure terminal
device(config)# router bgp
device(config-bgp-router)# local-as 65520
device(config-bgp-router)# no neighbor 2001:2018:8192::125 shutdown
```

Configure GTSM for BGP4+

You can configure GTSM to protect external Border Gateway Protocol (eBGP) peering sessions.

About This Task

For more information, see Generalized TTL Security Mechanism on page 202.

Procedure

1. Access global configuration mode.

```
device# configure terminal
```

2. Access BGP configuration mode.

```
device(config)# router bgp
```

3. Specify the autonomous system in which your device resides.

```
device(config-bgp-router)# local-as 65520
```

4. Specify the autonomous system in which the remote neighbor resides.

```
device(config-bgp-router)# neighbor 2001:2018:8192::125 remote-as 2
```

5. Enable GTSM between the device and the neighbor.

```
device(config-bgp-router)# neighbor 2001:2018:8192::125 ebgp-btsh
```

Example

The following example summarizes the commands in this procedure.

```
device# configure terminal
device(config)# 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
```

Disable the BGP4+ AS_PATH Check Function

When you disable the AS_PATH check function for routes learned from a specific location, BGP does not reject routes that contain the recipient BGP speaker's AS number.

Procedure

1. Access global configuration mode.

   ```
   device# configure terminal
   ```

2. Access BGP configuration mode.

   ```
   device(config)# router bgp
   ```

3. Access IPv6 address family unicast configuration mode.

   ```
   device(config-bgp-router)# address-family ipv6 unicast
   ```

4. Disable the BGP AS_PATH check function.

   ```
   device(config-bgp-ipv6u)# neighbor 2001:db8:e0ff:783a::4 allowas-in 3
   ```

   This example specifies the **neighbor allowas-in** command, the IPv6 address of the neighbor, and the variable *3*, which represents the number of times that the AS path of a received route can contain the BGP speaker's ASN and still be accepted.

Example

The following example summarizes the commands in this procedure.

```
device# configure terminal
device(config)# 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.

Before You Begin

About This Task
Use one or more of the following commands to verify BGP4+ information. The commands do not have to be entered in this order.

Procedure

1. Enter the **show ipv6 bgp summary** command.

```
device# show ipv6 bgp summary

  BGP4 Summary
  Router ID: 107.1.1.8   Local AS Number: 100
  Confederation Identifier: not configured
  Confederation Peers:
  Maximum Number of IP ECMP Paths Supported for Load Sharing: 1
  Number of Neighbors Configured: 1, UP: 0
  Number of Routes Installed: 1, Uses 96 bytes
  Number of Routes Advertising to All Neighbors: 1 (1 entries), Uses 60 bytes
  Number of Attribute Entries Installed: 1, Uses 104 bytes
  Neighbor Address  AS#          State     Time     Rt:Accepted Filtered Sent   ToSend
  1:2::3            100          CONN      0h 0m18s 0           0        0      1
```

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: 1
1       Next Hop  : ::                                    MED      :
0            Origin:INCOMP
        Originator:0.0.0.0          Cluster List:None
        Aggregator:AS Number :0          Router-ID:0.0.0.0        Atomic:None
        Local Pref:100             Communities:Internet
        AS Path   : (length 0)
           AsPathLen: 0  AsNum: 0, SegmentNum: 0, Neighboring As: 0, Source As 0
        Address: 0x0b456c4c  Hash:876 (0x03000000)
        Links: 0x00000000, 0x00000000
        Reference Counts: 1:0:1, Magic: 2
```

This example shows information about an route-attribute entry that is stored in device memory.

3. Enter the **show ipv6 bgp peer-group** command.

```
device# show ipv6 bgp peer-group

1   BGP peer-group is pg
      Address family : IPV4 Unicast
        activate
      Address family : IPV6 Unicast
        no activate
    Members:
       IP Address: 1.1.1.1, AS: 100
       IP Address: 1::1, AS: 100

2   BGP peer-group is pg6
      Address family : IPV4 Unicast
        activate
      Address family : IPV6 Unicast
        no activate
    Currently there are no members.
```

This example shows output for two peer groups, called "pg" and "pg6".

4. Enter the **show ipv6 bgp routes** command.

```
device# show ipv6 bgp routes

Total number of BGP Routes: 1
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     107:1:1::/64      ::              0          100         32768  BL
        AS_PATH:
```

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    : 1
  Distinct BGP destination networks               : 1
  Filtered bgp routes for soft reconfig           : 0
  Routes originated by this router                : 1
  Routes selected as BEST routes                  : 1
  Routes Installed as BEST routes                 : 1
  BEST routes not installed in IP forwarding table : 0
  Unreachable routes (no IGP route for NEXTHOP)   : 0
  IBGP routes selected as best routes             : 0
```

```
    EBGP routes selected as best routes             : 0
    BEST routes not valid for IP forwarding table   : 0
```

This example shows summarized BGP4+ route information.

Displaying BGP4+ neighbor statistics

Various **show ipv6 bgp neighbor** commands verify information about BGP4+ neighbor configurations.

Before You Begin

About This Task

Use one or more of the following commands to verify BGP4+ neighbor information. The commands do not have to be entered in this order.

Procedure

1. Enter the **show ipv6 bgp neighbors** command.

```
device# show ipv6 bgp neighbors

    Total number of BGP Neighbors: 1
1   IP Address: 1:2::3, AS: 100 (IBGP), RouterID: 0.0.0.0, VRF: default-vrf
    State: CONNECT, Time: 0h3m3s, KeepAliveTime: 60, HoldTime: 180
    Minimal Route Advertisement Interval: 0 seconds
    Messages:    Open    Update   KeepAlive Notification Refresh-Req
      Sent   : 0       0        0         0           0
      Received: 0       0        0         0           0
    Last Connection Reset Reason:Unknown
    Notification Sent:     Unspecified
    Notification Received: Unspecified
    Neighbor NLRI Negotiation:
      Peer configured for IPV6 unicast  Routes
    Neighbor ipv6 MPLS Label Capability Negotiation:
    Neighbor AS4 Capability Negotiation:
    Outbound Policy Group:
      ID: 2, Use Count: 3
      Last update time was 172 sec ago
Error: TCP status not available
```

This example output gives summarized information about a BGP4+ neighbor.

2. Enter the **show ipv6 bgp neighbors advertised-routes** command.

```
device# show ipv6 bgp neighbors 123::3 advertised-routes

      There are 5 routes advertised to neighbor 123::3
Status A:AGGREGATE B:BEST b:NOT-INSTALLED-BEST E:EBGP I:IBGP L:LOCAL
      Prefix            Next Hop        MED         LocPrf      Weight Status
1     110:110:110::/64   123::2          0                       0      BE
        AS_PATH: 222 111
2     110:110:110:1::/64 123::2          0                       0      BE
        AS_PATH: 222 111
3     110:110:110:2::/64 123::2          0                       0      BE
        AS_PATH: 222 111
4     110:110:110:3::/64 123::2          0                       0      BE
        AS_PATH: 222 111
5     110:110:110:4::/64 123::2          0                       0      BE
        AS_PATH: 222 111
```

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 neighbors 123::3 last-packet-with-error

 Received Message Length: 45
 BGP Message:
  0xffffffff  0xffffffff  0xffffffff  0xffffffff  0x002d0104
  0x014b00b4  0x09090909  0x10020601  0x04020000  0x01020202
  0x00020280  0x00

 BGP Header
  Marker:   0xffffffff  0xffffffff  0xffffffff  0xffffffff
  Message Length: (0x002d) 45
  Message Type: (0x01) OPEN

 OPEN Message
 Version: (0x04) 4
 AS Number: (0x014b) 331
 Hold Time: (0x00b4) 180
 BGP Identifier: (0x09090909) 9.9.9.9
 Optional Parameter length: (0x10) 16

 OPEN message optional parameters
  Parameter Type: (0x02) Capability
  Parameter Length: (0x06) 6
   Capability Type: (0x01) MULTIPROTOCOL EXTENSIONS
   Capability Length: (0x04) 4
   AFI: (0x0200) Unknown(512)
   Reserved: (0x00) 0
   SAFI: (0x01) Unicast

  Parameter Type: (0x02) Capability
  Parameter Length: (0x02) 2
   Capability Type: (0x02) ROUTE REFRESH(new)
   Capability Length: (0x00) 0

  Parameter Type: (0x02) Capability
  Parameter Length: (0x02) 2
   Capability Type: (0x80) ROUTE REFRESH(old)
   Capability Length: (0x00) 0
```

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 neighbors 160:160:160::10 received-routes

      There are 5 received routes from neighbor 160:160:160::10
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     110:110:110::/64   160:160:160::10  0          100       0      BE
        AS_PATH: 111
2     110:110:110:1::/64 160:160:160::10  0          100       0      BE
        AS_PATH: 111
3     110:110:110:2::/64 160:160:160::10  0          100       0      BE
        AS_PATH: 111
4     110:110:110:3::/64 160:160:160::10  0          100       0      BE
        AS_PATH: 111
5     110:110:110:4::/64 160:160:160::10  0          100       0      BE
        AS_PATH: 111
```

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 123::3 rib-out-routes

        There are 5 RIB_out routes for neighbor 123::3
Status A:AGGREGATE B:BEST b:NOT-INSTALLED-BEST E:EBGP I:IBGP L:LOCAL
        Prefix              Next Hop         MED        LocPrf      Weight Status
1       110:110:110::/64    160:160:160::10  0          100         0      BE
          AS_PATH: 111
2       110:110:110:1::/64  160:160:160::10  0          100         0      BE
          AS_PATH: 111
3       110:110:110:2::/64  160:160:160::10  0          100         0      BE
          AS_PATH: 111
4       110:110:110:3::/64  160:160:160::10  0          100         0      BE
          AS_PATH: 111
5       110:110:110:4::/64  160:160:160::10  0          100         0      BE
          AS_PATH: 111
```

This example shows information about BGP4+ outbound RIB routes.

Clearing BGP4+ dampened paths

BGP4+ suppressed routes can be reactivated using a CLI command.

Before You Begin

About This Task
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.

Procedure

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

    Status Code  >:best d:damped h:history *:valid
    Network
From                                      Flaps Since        Reuse       Path
*d  110:110:110:4::/64
160:160:160::10                           36    0 :2 :54    0 :10:10    111
*d  110:110:110:3::/64
160:160:160::10                           36    0 :2 :54    0 :10:10    111
*d  110:110:110:2::/64
160:160:160::10                           36    0 :2 :54    0 :10:10    111
*d  110:110:110:1::/64
160:160:160::10                           36    0 :2 :54    0 :10:10    111
*d  110:110:110::/64
160:160:160::10                           36    0 :2 :54    0 :10:10    111
```

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#
```

Example

The following example reactivates all suppressed BGP4+ routes and verifies that there are no suppressed routes.

```
device(config-bgp-router)# exit
device(config)# exit
device# show ipv6 bgp dampened-paths
device# clear ipv6 bgp dampening
device# show ipv6 bgp dampened-paths
```

BGP EVPN for IP Fabrics

BGP EVPN Overview

Ethernet VPN (EVPN) provides a standards-based solution for data center overlay and data center interconnect (DCI).

RFC 7432 ("BGP MPLS-Based Ethernet VPN") specifies multiprotocol extensions to BGP to exchange Layer 2 routes in an MPLS Ethernet VPN network. These extensions are useful in DCI scenarios.

A BGP EVPN-based IP Fabric consists of BGP EVPN for VXLAN overlay networks and a broad set of Layer 2, Layer 3, and infrastructure features to enable seamless deployment, on-demand usage of forwarding entries in hardware, and minimization of flooding in the network.

IP Fabrics topologies

The following figure shows a supported topology where the SLX devices are nodes in a BGP EVPN spine in an IP Fabrics network.
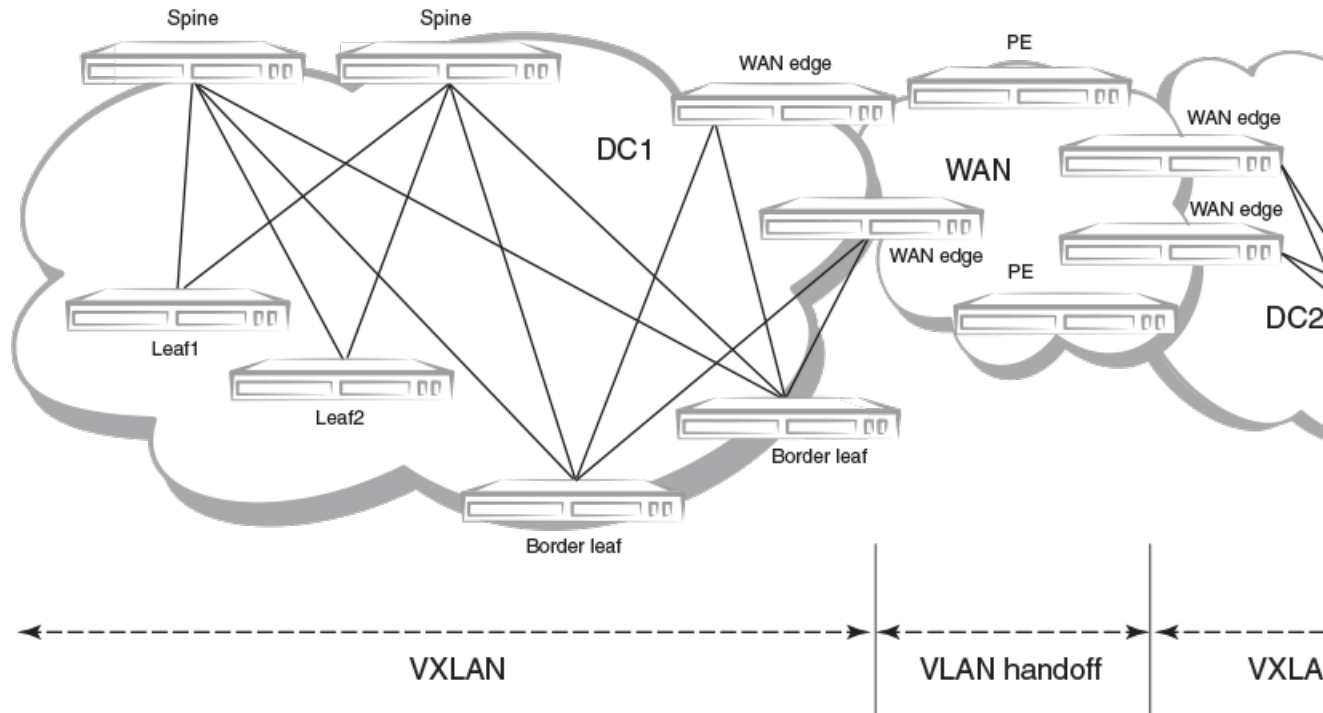
Figure 23: Supported BGP EVPN spine topology

The leaf nodes in the IP Fabric can be any of the supported hardware platforms. A BGP EVPN-based Multi-Chassis Trunk (MCT) solution is also supported on those platforms. However, using a co-located BGP EVPN spine and a BGP EVPN MCT on the same switch is not supported. For a list of supported platforms, see Supported Hardware on page 22.

VXLANs provide the inter-connectivity between leaf nodes in an IP Fabric network. VXLAN tunnels are established between leaf nodes. The following figure illustrates this VXLAN connectivity.
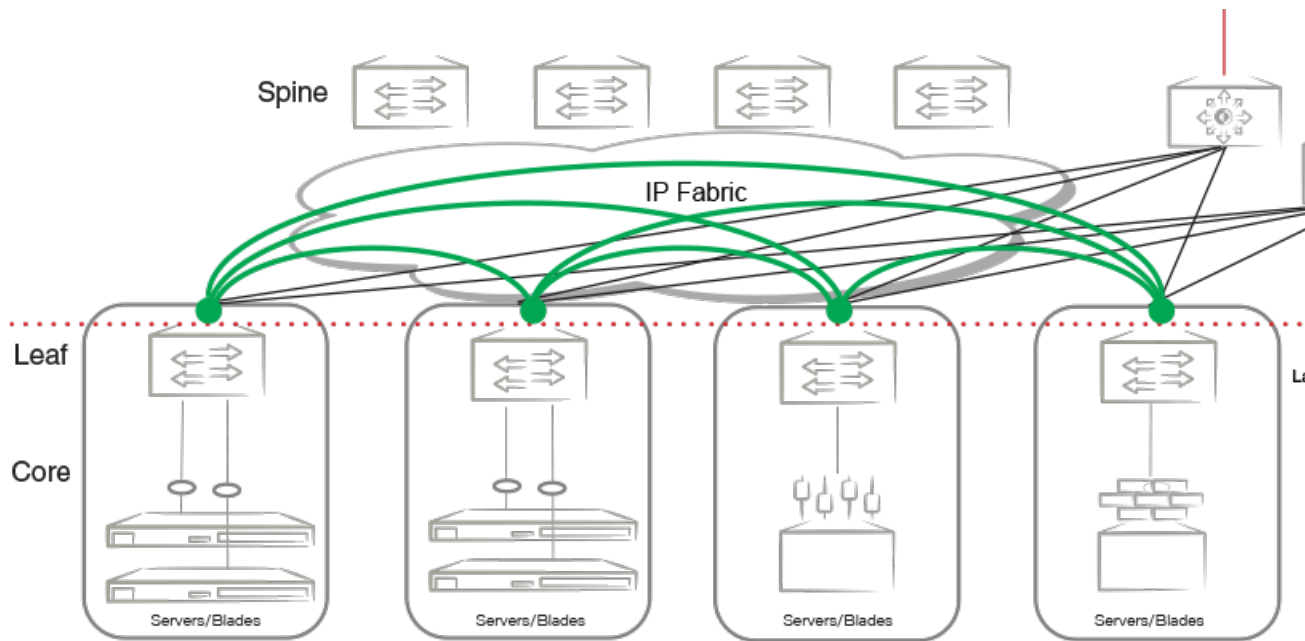
Figure 24: VXLAN connectivity in an IP Fabric

> **Note**
> Leaf nodes will be dual-homed.

Supported IP Fabrics features

- BGP EVPN spine with route reflector control plane
- VXLAN VTEP discovery
- VXLAN logical VTEP
- BGP EVPN-based MCT cluster formation
- Layer 2 (MAC) route exchange
- ARP/ND (MACIP) route exchange
- Static anycast gateway
- ARP suppression
- IP unnumbered interfaces
- Multi-VRF support and EVPN prefix route exchange
- Conversion of MACIP routes to host prefix routes

BGP EVPN use cases

- BGP EVPN VXLAN overlay in the data center
- MCT cluster formation
- VXLAN data-center interconnect (Layer 2 services on the SLX 9540 only)

BGP EVPN control plane

This section discusses the following components of the BGP EVPN control plane.

Supported route types

The following table describes the supported EVPN route types.

Table 26: Supported EVPN route types

Type	Description
1	Ethernet Auto Discovery (AD) route
2	MAC/IP advertisement route
3	Inclusive Multicast Route
4	Ethernet Segment Route
5	IPv4/IPv6 Prefix Route
7	IGMP Join Synch Route
8	IGMP Leave Synch Route

Supported BGP functionalities

The following features are supported for BGP EVPN address-family:

- iBGP and eBGP peering
- IPv4 and IPv6 peer types
- Peer groups with EVPN neighbors
- Ability to negotiate only EVPN address-family for IPv4/IPv6 peers
- iBGP route reflector server
- Keeping next-hop unchanged for EBGP peers
- Ability to retain all EVPN routes without importing them
- Coexistence of all IPv4 and IPv6 BGP features for neighbors negotiating EVPN address-family
- EVPN spine, leaf, and border leaf functionalities

Supported data plane encapsulation types

The following data-plane encapsulation types are supported:

- MPLS (for MCT support)
- VXLAN

MPLS is the default encapsulation type.

The BGP encapsulation extended-community attribute is carried with each route in the BGP control plane to signify the encapsulation to be used to reach the prefix. An EVPN route not carrying any encapsulation extended-community attribute implicitly means that MPLS is encapsulation type.

If an EVPN prefix is received from multiple sources with both VXLAN and MPLS encapsulations, the route source with VXLAN encapsulation is preferred within the data center.

Supported service-interface model

The VLAN-based service-interface model described in RFC 7432 is supported. Each VLAN is mapped to a unique VNI/MPLS label. Within the data center, each VNI maps to a unique EVI; this mapping is referred to as the "single subnet per EVI" option in "draft-ietf-bess-evpn-overlay." No VLAN translation service is supported. Within the data center, VNI-to-VLAN mapping is local to each leaf, and the inner VXLAN frames may not carry an Ethernet tag. The administrator is responsible for keeping this mapping consistent within the data center.

BGP EVPN configuration examples

BGP EVPN neighbor examples

This section presents configuration examples of basic BGP EVPN neighbor configurations.

Configuring a BGP EVPN neighbor and peer group

EVPN address-family can be configured to be negotiated on either IPv4 or IPv6 BGP neighbors, as shown in the following example.

```
device(config)#router bgp
device(config-bgp-router)# local-as 62001
device(config-bgp-router)# neighbor 192.168.21.31 remote-as 62000
device(config-bgp-router)# neighbor 2000:10:1::2 remote-as 62002
device(config-bgp-router)# address-family l2vpn evpn
device(config-bgp-evpn)# retain route-target all <---Spine only
device(config-bgp-evpn)# neighbor 192.168.21.31 next-hop-unchanged <---Spine only
device(config-bgp-evpn)# neighbor 192.168.21.31 activate
device(config-bgp-evpn)# neighbor 2000:10:1::2 next-hop-unchanged <---Spine only
device(config-bgp-evpn)# neighbor 2000:10:1::2 activate
device(config-bgp-evpn)# end
```

As with any VPN technology, EVPN neighbors must always be in the default VRF. The preceding example shows EVPN address-family being activated for IPv4 and IPv6 BGP neighbors.

BGP neighbor configuration can be simplified by using peer groups, as in the following example.

```
device(config-bgp-router)# neighbor my-peer-group peer-group
device(config-bgp-router)# neighbor my-peer-group remote-as 62001
device(config-bgp-router)# neighbor 192.168.30.2 peer-group my-peer-group
device(config-bgp-router)# neighbor 2000:20:1::2 peer-group my-peer-group
device(config-bgp-router)# address-family l2vpn evpn
device(config-bgp-evpn)# retain route-target all
device(config-bgp-evpn)# neighbor my-peer-group route-reflector-client
device(config-bgp-evpn)# neighbor my-peer-group activate
```

> **Note**
> The **neighbor route-reflector client** command is not required if all devices peer with each other in a full iBGP mesh.

Specifying BGP EVPN neighbor encapsulation

BGP EVPN routes carry the encapsulation type as part of the BGP encapsulation extended community. When BGP encapsulation extended community is absent from the route, the default MPLS encapsulation is assumed. On SLX-OS platforms, the default encapsulation for EVPN peers is MPLS, and BGP encapsulation extended community with the MPLS encapsulation type is attached to all EVPN routes.

Retaining EVPN routes without importing them

For spine or specific border leaf cases, routes are not imported into MAC-VRF or IP-VRF tables, but instead are reflected/re-advertised to iBGP/eBGP peers. The following configuration is required to retain all of the routes in the VPN table.

```
device(config-bgp-router)# address-family l2vpn evpn
device(config-bgp-evpn)# retain route-target all
```

Keeping next-hop unchanged for eBGP neighbors

When BGP advertises routes to an eBGP peer, the next-hop value in the route is modified to carry the local peering IP address. In the case of a BGP EVPN VXLAN IP Fabric, the next-hop value in the route is the VTEP IP address of the advertising router. A BGP EVPN spine distributing the routes to other leaf nodes should not modify the next-hop value. Each EBGP peer activated under EVPN address-family should be configured so that it does not modify the next-hop value, as in the following example.

```
device(config-bgp-router)# address-family l2vpn evpn
device(config-bgp-evpn)# neighbor 192.168.21.31 next-hop-unchanged
device(config-bgp-evpn)# neighbor my-peer-group next-hop-unchanged
device(config-bgp-evpn)# neighbor 192.168.21.31 activate
device(config-bgp-evpn)# neighbor my-peer-group activate
```

Allowing routes with a local AS number

In special cases where spine or super-spine routers must accept EVPN routes that have the AS number of a router in the AS path segment, the **allowas-in** command can be used for a given BGP peer, as in the following example.

```
device(config-bgp-router)# address-family l2vpn evpn
device(config-bgp-evpn)# neighbor 192.168.21.31 allowas-in 1
device(config-bgp-evpn)# neighbor my-peer-group allowas-in 1
```

Configuring iBGP EVPN neighbors

In a full iBGP mesh of EVPN nodes, a leaf node needs to peer only with all other leaf nodes; it does not need to have a session with any spine node. The configuration of route-reflector client on the spine nodes is not required.

In a non-full iBGP mesh configuration, in order to reflect routes from one iBGP neighbor to another iBGP neighbor, the iBGP neighbors should be configured as route-reflector-clients, as in the following example.

```
device(config-bgp-router)# address-family l2vpn evpn
device(config-bgp-evpn)# neighbor 192.168.21.31 route-reflector-client
device(config-bgp-evpn)# neighbor my-peer-group route-reflector-client
```

The '"client-to-client-reflection" feature is by default enabled under EVPN address-family. The route-reflector configuration does not force routes that are not imported to be retained in BGP. The **retain route-target all** command is required in all cases.

Filtering EVPN routes

Limited filtering support is available for EVPN routes. However, route maps can be configured on EVPN neighbors, as in the following example.

```
device(config-bgp-router)# address-family l2vpn evpn
device(config-bgp-evpn)# neighbor 192.168.21.31 route-map in rmap-in
device(config-bgp-evpn)# neighbor 192.168.21.31 route-map out rmap-out
```

Only AS path and extended-community-based match and set criteria are supported for EVPN routes.

Negotiating only EVPN address-family

In order to have better resiliency, separate BGP neighbors for overlay and underlay may be desired. This separation allows administrators to manipulate, debug, and maintain EVPN neighbors without affecting underlay connectivity. This separation can be achieved by deactivating underlay address-family for EVPN neighbors.

IPv4 neighbors are automatically activated under IPv4 unicast address-family, and they must be deactivated as in the following example.

```
device(config-bgp-router)# neighbor 192.168.21.31 remote-as 62000
device(config-bgp-router)# address-family ipv4 unicast
device(config-bgp-ipv4u)# no neighbor 192.168.21.31 activate
device(config-bgp-ipv4u)# address-family l2vpn evpn
device(config-bgp-evpn)# neighbor 192.168.21.31 activate
```

IPv6 neighbors are not by default activated under IPv6 unicast address-family. Therefore, for them to negotiate only EVPN address-family, they must be activated only under EVPN, as in the following example.

```
device(config-bgp-router)# neighbor 2000:10:1::2 remote-as 62002
device(config-bgp-ipv4u)# address-family l2vpn evpn
device(config-bgp-evpn)# neighbor 2000:10:1::2 activate
```

Enabling peer AS check

When a BGP router receives an update message with its own AS number in the AS path attribute, this means that there is an AS path loop and by default such updates are discarded. In many scenarios this AS path check needs to be disabled, and seeing a the AS number of a router in the AS path attribute is fine. On SLX devices, this AS path check is enforced at the receiver and is overridden by means of the **neighbor allowas-in** command. However, enforcing this AS path check at the sender side can save the amount of RIBout memory used to store the routes that are discarded at the receiver. Furthermore, it avoids sending updates and withdraws those routes, thereby improving the convergence time.

The following example configuration shows how to enforce this check for a neighbor or peer group. This configuration is per address-family.

```
LeafC_2(config-bgp-router)# address-family ipv4 unicast
LeafC_2(config-bgp-ipv4u)# neighbor 163.124.20.10 enable-peer-as-check
```

A neighbor reset is required for the preceding configuration to take effect. After it is applied, this configuration prevents Network Layer Reachability Information (NLRI) from being added to the RIBout of the peer if the AS path segment of the NLRI has the AS number configured as "remote-as" for the neighbor.

BGP EVPN instance configuration

With a VLAN-based service model, each VLAN/BD corresponds to a unique EVPN instance, or MAC-VRF. Each route in EVPN belonging to a MAC-VRF has a unique route distinguisher (RD) that differentiates routes from other MAC-VRFs.

In addition, sets of route targets (export RTs) are associated with each MAC-VRF; these RTs are applied to the routes while advertising. RTs in the route are matched against the configured import RTs. If any of the RTs match, the route is imported; otherwise, it is discarded.

> **Note**
> Auto RD is mandatory for EVI extension with manual RD and RT/auto RT.

Each VLAN/BD to be extended into an EVPN overlay must be added under the EVPN configuration block, as shown in the following configuration example.

```
evpn t1
!
vlan 1020
  rd 1020:1
  route-target both 1020:100
!
vlan 3200
 rd 3200:1
 route-target both 3200:100
!
bridge-domain 10
 rd 10:1
 route-target both 10:100
!
bridge-domain 3100
 rd 3100:1
 route-target both 3100:1
```

Each VLAN/BD added to the preceding EVPN configuration is considered an EVI and is assigned a unique EVID internally. EVIDs for VLANs/BDs are calculated as shown in the following table.

Table 27: Calculating EVIDs for VLANBs/BDs

VLAN/BD	EVI value
VLAN (1–4095)	VLANID
BD (1–4096+)	4096 + BD-ID

The EVID value in the preceding configuration example is not sent in the Ethernet-tag field in the routes. It is always 0 for VLAN-aware service. However, routes imported into the MAC-VRF table are classified against an EVID, which is shown in the Ethernet-tag field.

Autogeneration of route distinguisher and route target

Because thousands of VLANs/BDs can exist in the network, configuring each of them individually on each router is cumbersome. In order to simplify the configuration, Extreme recommends the autogeneration of RDs and RTs.

The RD of EVPN routes is encoded as a Type-1 route-distinguisher, as described in RFC4364. The following figure shows the formats of RD and BGP RT extended community.
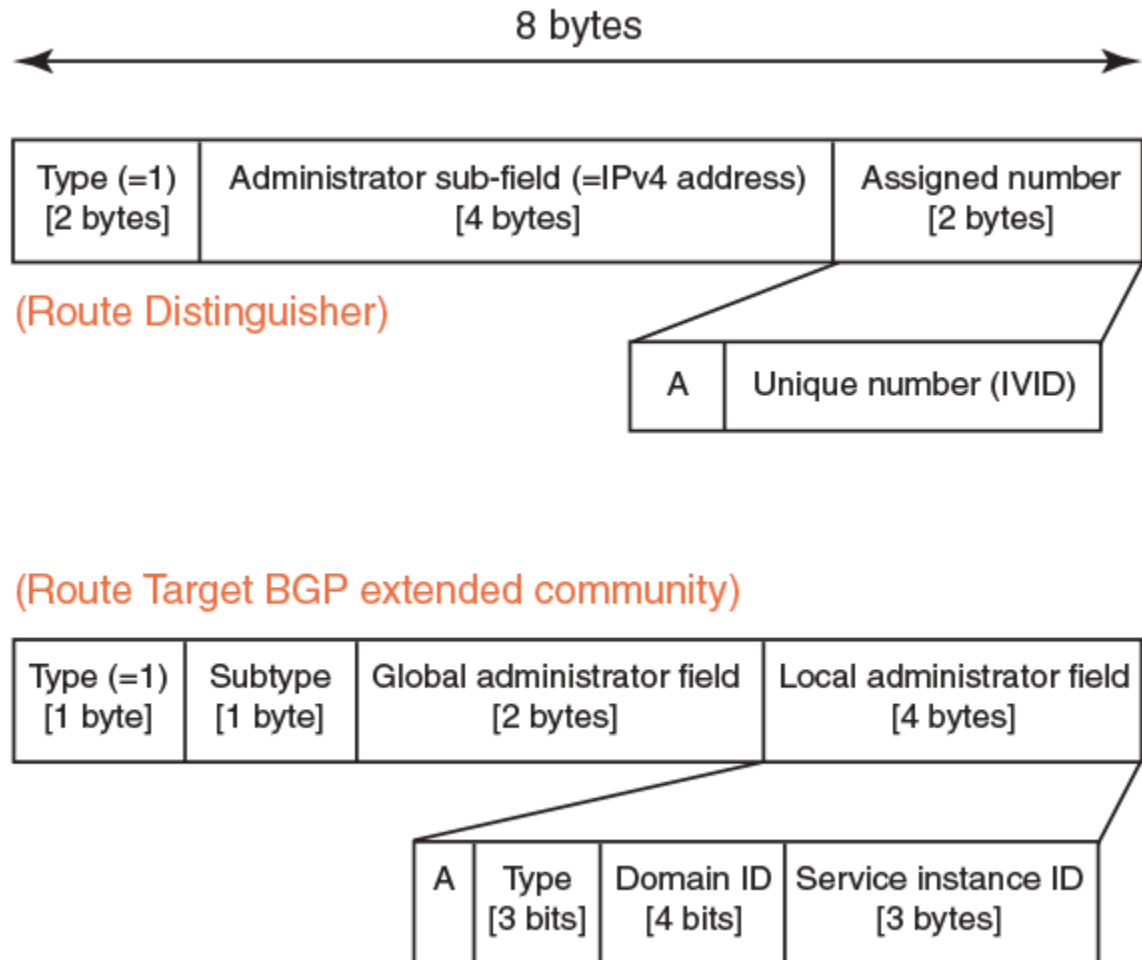
Figure 25: RD and RT formats

The most significant bit "A" of the assigned number field stands for automatic or manual RD. For a manual RD value, this bit is set and the unique identifier value is set to the internal VLAN ID (IVID) to keep the RD unique across VLANs and bridge domain IDs (BD-IDs). The administrator subfield of the RD is set to the BGP router ID.

The global administrator field in the RT is set to the AS number. The local administrator field in the RT is expanded, as shown in the preceding figure. Bit-value "A" stands for automatic or manual RT. Where the RT value is derived automatically, this bit is set to 0. The service instance ID field carries the VLAN or VNI value. The Type field is set, according to the value set in the service instance ID space, to either 0 (= IVID) or 1 (= VNI). The domain-id value is not used currently and is set to 0.

RTs with the following service instance types are attached according to neighbor encapsulation:

Table 28: Neighbor encapsulation and RT service instance types

Neighbor encapsulation	RT service instance type
VXLAN	VXLAN (1)
MPLS	EVID (4)

The configuration of VLANs/BDs that use autogenerated RD and RT values is simplified, as shown in the following configuration example.

```
evpn
 rd auto
 route-target both auto ignore-as
 vlan add 1000-2000
 bridge-domain add 1000-2000
```

VLAN and bridge domains that are added with range, RD, and RT values are specified as "auto". In scenarios where multiple leaf nodes belong to different autonomous systems, Extreme recommends the "ignore-as" option. When that option is specified, the AS number field in the RT in the route is not compared against the local AS number.

BGP dynamic neighbors for an IP Fabric

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.

In a typical data center deployment, a spine has a BGP session with each of the leaf nodes. Typically in this situation, the ratio of spine nodes to leaf nodes is 1 to 4. The greater the number of leafs, the greater the administrative overhead at the spine. Every time a new leaf node is added, specific configurations must be added for each spine node. Moreover, if the IP address used by a leaf for neighboring with the spine node changes, BGP peering information at all spines must be changed. Therefore, as the number of leaf node connections increases, a great deal of administrative work can be required.

The following figure illustrates a typical data center deployment where a spine has a BGP session with each of the individual leaf nodes.
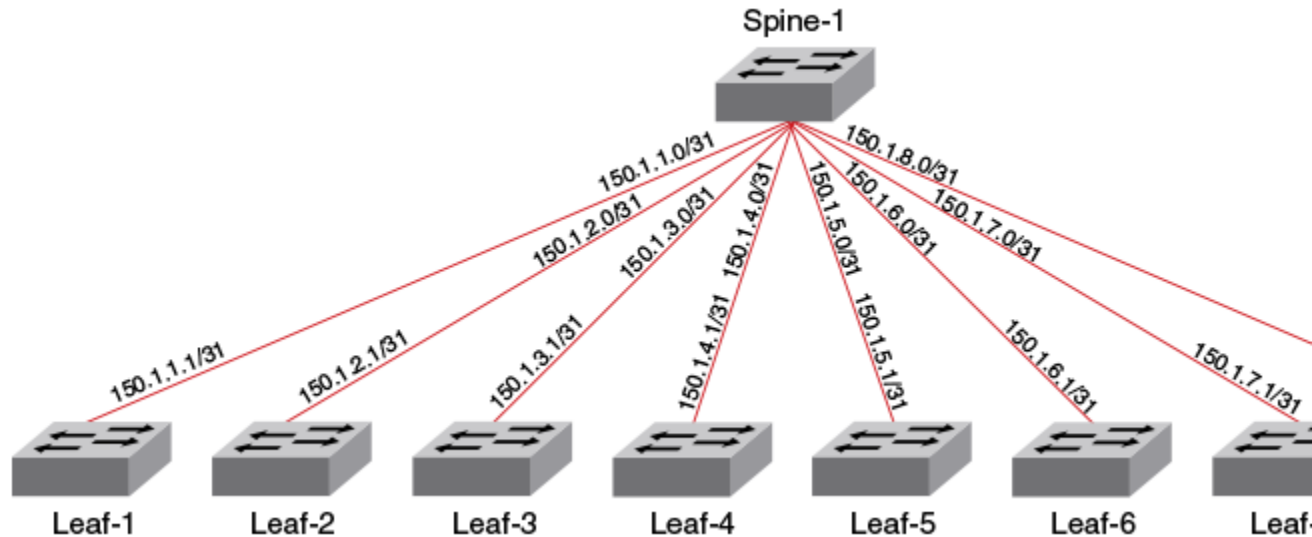
Figure 26: BGP session between spine and leaf nodes

In order for the spine to establish sessions with each individual leaf node, as depicted in the previous figure, a great deal of configuration is needed at the spine level. The following configuration is required at the spine level to establish such a session with the leaf nodes.

```
device# configure terminal
device(config)# router bgp
device(config-bgp-router)# local-as 65500
device(config-bgp-router)# neighbor 150.1.1.1 remote-as 65550
device(config-bgp-router)# neighbor 150.1.2.1 remote-as 65550
device(config-bgp-router)# neighbor 150.1.3.1 remote-as 65550
device(config-bgp-router)# neighbor 150.1.4.1 remote-as 65550
device(config-bgp-router)# neighbor 150.1.5.1 remote-as 65550
device(config-bgp-router)# neighbor 150.1.6.1 remote-as 65550
device(config-bgp-router)# neighbor 150.1.7.1 remote-as 65550
device(config-bgp-router)# neighbor 150.1.8.1 remote-as 65550
device(config-bgp-router)# address-family l2vpn evpn
device(config-bgp-evpn)# neighbor 150.1.1.1 activate
device(config-bgp-evpn)# neighbor 150.1.2.1 activate
device(config-bgp-evpn)# neighbor 150.1.3.1 activate
device(config-bgp-evpn)# neighbor 150.1.4.1 activate
device(config-bgp-evpn)# neighbor 150.1.5.1 activate
device(config-bgp-evpn)# neighbor 150.1.6.1 activate
device(config-bgp-evpn)# neighbor 150.1.7.1 activate
device(config-bgp-evpn)# neighbor 150.1.8.1 activate
```

With BGP dynamic neighbors, the previous configuration is reduced to the following configuration.

```
device# configure terminal
device(config)# router bgp
device(config-bgp-router)# local-as 65500
device(config-bgp-router)# neighbor pg1 peer-group
device(config-bgp-router)# neighbor pg1 remote-as 65550
device(config-bgp-router)# listen-range 150.1.0.0/16 peer-group pg1 limit 8
device(config-bgp-router)# address-family l2vpn evpn
device(config-bgp-evpn)# neighbor pg1 activate
```

When BGP dynamic neighbors is enabled for a spine, a BGP subnet listen range (a subnet range address that the IP address of all connected leafs falls under) is configured. Leaf switches are added as

BGP peers based on the listen range. The following figure illustrates the minimum configuration needed for a single spine to enable BGP dynamic neighbors.
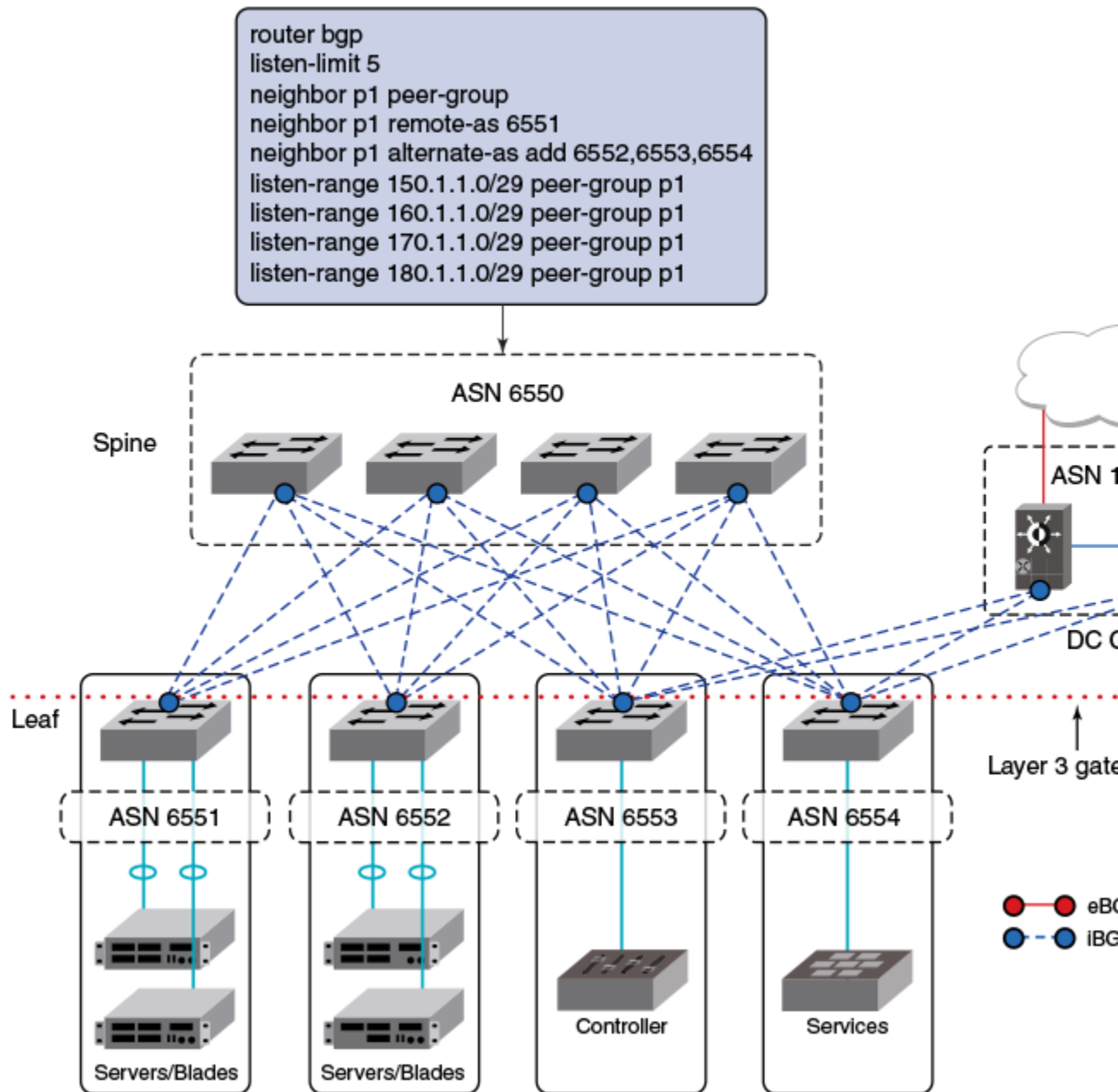
Figure 27: Minimum configuration at the spine level for BGP dynamic neighbors

When eBGP peers have similar configurations, the remote AS number (ASN) cannot be configured as part of the peer group because each peer has its own AS number. For static eBGP peers, a remote ASN must first be configured and then attached to the peer group. For BGP dynamic neighbors, you can configure an alternate AS range; an AS range under which the eBGP peer can fall for the listen range the peer group is a part of. This alternate range is used only for BGP dynamic neighbors. The previous figure illustrates the configuration for the deployment of dynamic eBGP neighbors.

When an OPEN message is received from a particular peer with an AS number that falls under the range configured for the peer group, it is accepted and a session is formed.

For general information on BGP dynamic neighbors, refer to the *BGP4* chapter.

Configuring BGP dynamic neighbors for an IP Fabric

The following task is configured on a spine and creates a peer group called "leaf-group" and associates it with the listen range of 150.1.0.0/16. The spine waits for the incoming TCP session initiated by a leaf node. When the TCP connection is accepted, the spine verifies whether the incoming TCP session falls under one of the configured listen ranges. If the verification is successful, a new BGP dynamic peer is created and associated with the peer group "leaf-group". This newly created BGP dynamic peer inherits the attributes associated with the peer group.

Procedure

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

```
device# configure terminal
```

2. Enter the **router bgp** command to enable BGP routing.

```
device(config)# router bgp
```

3. Enter the **local-as** command to configure the autonomous system number (ASN) in which your device resides.

```
device(config-bgp-router)# local-as 65520
```

4. Enter the **neighbor peer-group** command, specifying a name for the peer group, to create a peer group.

```
device(config-bgp-router)# neighbor leaf-group peer-group
```

5. Enter the **listen-limit** command and specify a value to define the maximum number of BGP dynamic subnet range neighbors that can be created.

```
device(config-bgp-router)# listen-limit 30
```

6. Enter the **neighbor** *peer-group* **remote-as** command to specify an AS for the peer group.

```
device(config-bgp-router)# neighbor leaf-group remote-as 65550
```

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 leaf-group 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 leaf-group alternate-as add 1210
```

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* **activate** command to enable the exchange of information with the peer group.

```
device(config-bgp-evpn)# neighbor leaf-group activate
```

Example

The following example creates a peer group called "leaf-group" and associates it with the listen range of 150.1.0.0/16. It also sets a limit on the number of dynamic BGP neighbors that can be created. An alternate AS number of listen range neighbors in the configured peer group is set.

```
device# configure terminal
device(config)# router bgp
device(config-bgp-router)# local-as 65520
device(config-bgp-router)# neighbor leaf-group peer-group
device(config-bgp-router)# listen-limit 30
device(config-bgp-router)# neighbor leaf-group remote-as 65550
device(config-bgp-router)# listen-range 150.1.0.0/16 peer-group leaf-group limit 20
device(config-bgp-router)# neighbor leaf-group alternate-as add 1210
device(config-bgp-router)# address-family l2vpn evpn
device(config-bgp-evpn)# neighbor leaf-group activate
```

BGP routing tables

The EVPN table holds VPN routes belonging to the L2VPN EVPN address-family. These routes may be received from an EVPN peer, originated locally, or exported from other VRFs for prefix routes.

EVPN route types 1 through 4 received from remote peers are imported only into the MAC-VRF table, except for MACIP host routes that may get converted to IPv4/v6 host routes and imported into the IP-VRF table. EVPN routes received from remote peers are validated against import rules and are added to

the VPN table only if validation passes. The following table lists the import rules for peer types and route types.

Table 29: Import rules for peer and route types

Received from peer type	Route type	Import rules
VXLAN peer	MAC/IP, Ethernet Tag	RTs in the route are looked up in the MAC-VRF table. A route is imported for the following conditions: 1. RT in the route matches RTs configured for VLAN/BD. 2. When a VLAN/BD instance is not configured manually, a VLAN/BD must exist in the system and be added to EVPN. 3. L2VNI in the label field is same as corresponding VNI for the VLAN/BD. For MAC-IP (ARP/ND) routes, if no match is found in MAC-VRF, RTs in the route are looked up in IP-VRFs. For each IP-VRF matching the RT, the route is imported in that IP-VRF.
	IP Prefix	RTs in the route are looked up in IP-VRFs. For each IP-VRF matching the RT, route is imported into that IP-VRF.
	Ethernet Segment	Route is imported if the RT in the route matches the auto ES-RT of the locally configured MCT interface.
	Inclusive Multicast Routes	

The EVPN table can hold routes even if they are not imported into MAC-VRF or IP-VRF tables. This table is required at the spine or border-leaf nodes to reflect the routes to other leaf nodes or to other data centers, respectively. The `retain route-target all` command is used to configure this table.

The consolidation of the routes from different sources (RDs) occurs in the MAC-VRF table after the routes pass through route-target checking and import filtering. The following operations are performed exclusively in the MAC-VRF table:

- VTEP discovery
- MAC move detection
- MAC dampening
- ES-based route use

The following table lists advertisement behavior by route type.

Table 30: Advertisement behavior by route type

Route type	Advertised to VXLAN peer?	Advertised to MPLS peer?
Autodiscovery	No	Yes
MAC	Yes (without ESI)	Yes
MACIP (ARP/ND)	Yes (without ESI)	Yes
Inclusive Multicast	Yes	Yes
Ethernet Segment (ES)	No	Yes
IP-Prefix (IPv4Prefix/IPv6Prefix)	Yes	Yes (without router MAC)

BGP EVPN-based MCT cluster formation

Multi-Chassis Trunking (MCT) cluster formation leverages BGP EVPN Ethernet Segment and Auto-Discovery (AD) routes and employs proprietary mechanisms to avoid loops in the cluster in transient configuration scenarios.

No separate EVPN instance for MCT and non-MCT services exists. Therefore, to configure MCT even on a stand-alone MCT cluster, you need to configure EVPN.

Dependency on the EVPN instance

EVPN configuration is required (along with a BGP neighbor and an MCT cluster/client) to form an MCT cluster. The MCT cluster automatically becomes a member of the VLAN or bridge domain configured under EVPN. The following is an example configuration.

```
device(config)# cluster MCT1
device(config-cluster-MCT1)#

evpn
  rd auto
  route-target both auto ignore-as
  vlan add 100,200,1000-4000
  bridge-domain add 100,200,1000-4000
```

This configuration creates an EVPN configuration instance with auto RD and RT and adds a specified VLAN and bridge domain to the EVPN configuration.

BGP EVPN peering for the MCT cluster

MCT cluster formation is controlled by BGP EVPN peering between the cluster members. The following figure illustrates typical MCT use cases.
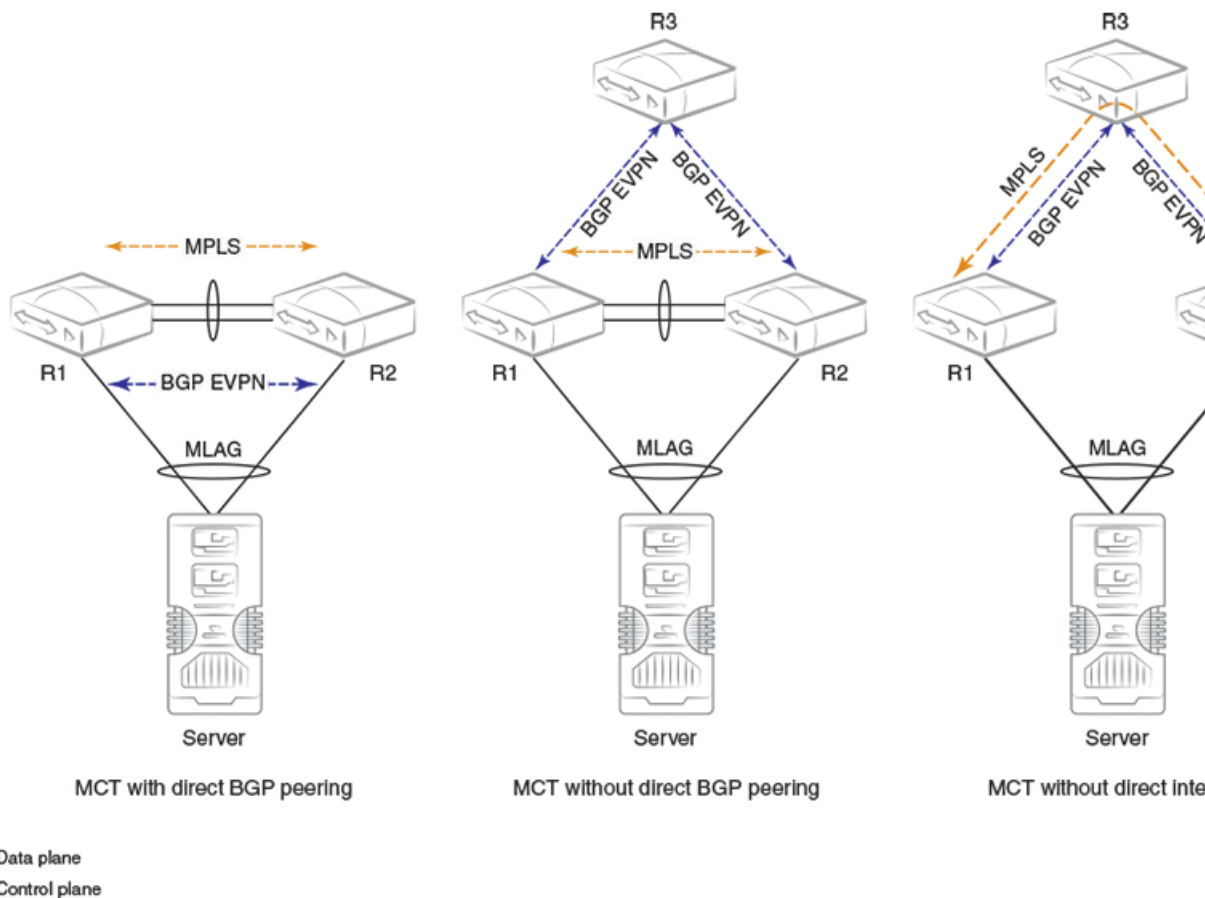
Figure 28: MCT use cases

BGP EVPN peering (not necessarily direct) between the MCT cluster members is required.

The first diagram shows a standalone MCT cluster and is the simplest MCT use case, where BGP EVPN peering exists only between the cluster members. An MPLS tunnel is established over the direct link (peer-interface), and the control and data planes are established over the same path.

The second diagram shows separate control and data plane paths. BGP EVPN peering is not direct between the cluster members. On SLX devices, the data plane is established using a VxLAN tunnel between MCT peers.

The third diagram shows MCT cluster formation without a direct link (peer interface). In this case, a multihop MPLS tunnel is established through intermediate routers.

The MCT cluster state is no longer dependent on BGP peering with the peer IP address specified under cluster configuration. Instead, the reachability of the BGP EVPN next-hop of the corresponding encapsulation is used for the cluster state.

Handling of ES/AD routes in BGP

ES and AD-per-ES routes are exchanged between MCT peers for Designated Forwarder (DF) election. The ES route is originated when the MCT cluster client is deployed. A set of AD-per-ES routes is originated when the cluster client interface is operationally up.

ES routes carry BGP ES-import route-target extended community, which is automatically constructed from the ESI value. ES-import RTs are automatically added in EVPN whenever a cluster client interface with an ES is configured. An ES route with an ESI value is imported only by those routers that have membership of that ES.

Similar to other EVPN routes, ES/AD routes are added to BGP RIB-in and are advertised to EVPN neighbors.

AD-per-ES routes are required to carry the route-targets of all EVIs that are members of the ES (cluster client interface). Because a large number of EVIs can be members of the ES (4K VLAN + BDs), the AD route may not fit in a single BGP update message, which is only 4K. Multiple AD-per-ES routes with unique RDs are generated for each ES as shown in the following table.

Table 31: RD index and EVI RTs

RD index for RD (Router-ID: RD-index)	RTs of the EVIs attached to AD-per-ES route
1	VLAN 1–127
2	VLAN 128–255
3	VLAN 256–383
. . .	. . .
32	VLAN 3968–4095
33	BD 1–127
34	BD 128–255
35	BD 256–383
...	...

AD-per-ES routes carry the ESI label in ESI label extended community for split-horizon filtering.

BGP EVPN-based VXLAN Overlay

BGP EVPN provides control plane signaling and tunnel discovery support in a VXLAN overlay network.

Inclusive multicast routes provide for the extension of VLANs and bridge domains over tunnels, and MAC, MACIP, and prefix routes provide for signaling control plane routes for tenants. Forwarding across the tenants or toward the core takes place through VXLAN overlay tunnels. The flooding of BUM traffic is achieved by means of ingress replication.

The following points apply to a BGP EVPN-based VXLAN overlay network:

- A mesh of tunnels from everywhere to everywhere is required in the data center.
- Flooding is achieved through Ingress replication. Multicast support does not exist.
- Spine nodes provide load balancing for overlay tunnels and alternate route sources for BGP.

Devices in the underlay Layer 3 Clos network may have the following distinct roles:

- Leaf
- Spine

- Border leaf
- Co-located border leaf and spine

Underlay architectures

The underlay architecture in a Layer 3 Clos network can be based on iBGP or eBGP. The following topologies illustrate these options.

iBGP spine and leaf

With iBGP-based underlay, all spine and leaf nodes are in the same AS. Spine nodes act as route-reflectors and do not terminate any tunnel, whereas leaf nodes originate and terminate all of the tunnels and routes. The following figure illustrates this topology.
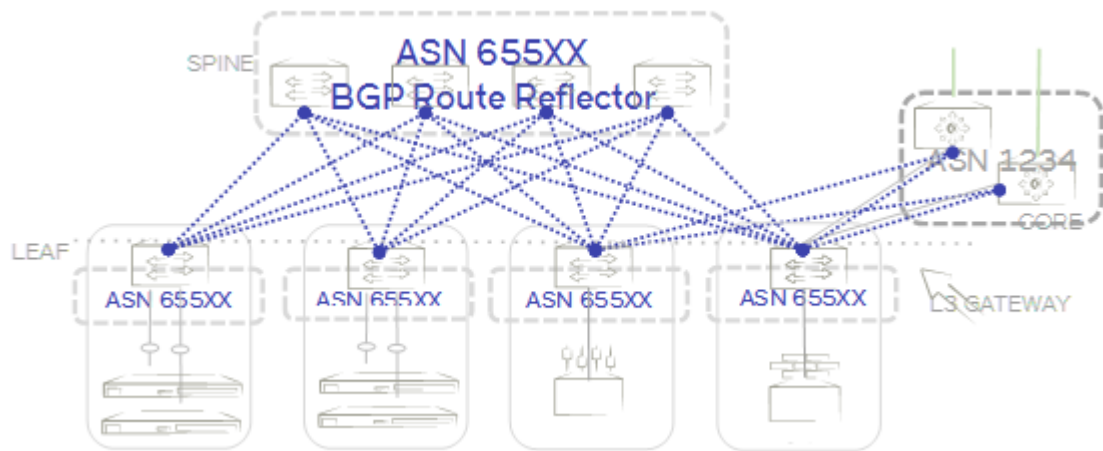
Figure 29: iBGP spine and leaf

eBGP spine and leaf

With eBGP-based underlay, each leaf node is assigned a distinct AS number. Usually, the iBGP underlay architecture provides higher BGP scalability but with less control on the routes getting exchanged. On the other hand, eBGP underlay provides lower BGP scalability but with greater control on routes, as policy and filters can be applied on the originating AS numbers. The following figure illustrates this topology.
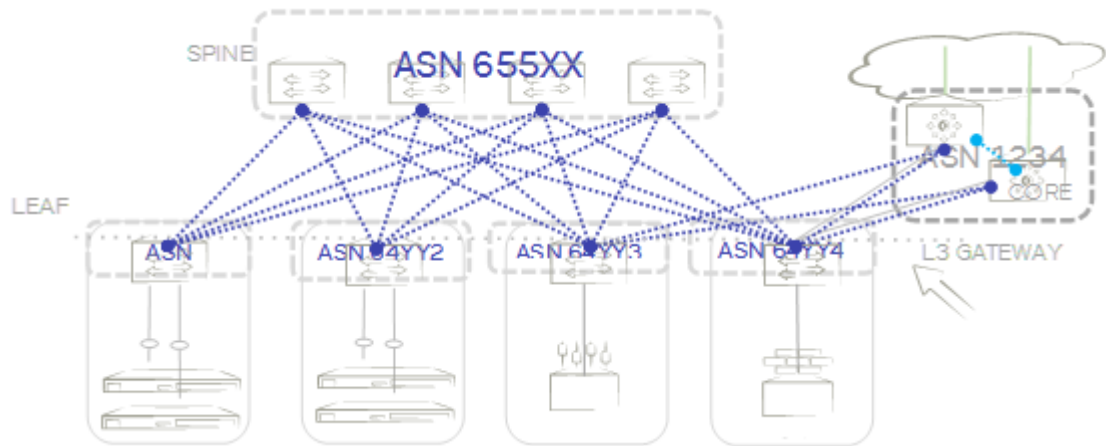
Figure 30: eBGP spine and leaf

Border and service leaf

A border leaf is special leaf node, typically redundant, that sits at the edge of the data center and provides termination, hand-off, or control-plane extension towards the WAN or core. Similarly, that functionality is provided in the reverse direction, from the WAN/core toward the data center.

A border leaf may act as a transparent route-reflector/forwarder (iBGP/eBGP) or may terminate tunnels while providing reflection service. In the service-leaf model, a border leaf also acts as just another leaf node in the network and may be extending Layer 2, Layer 3, or both services as illustrated in the following figure.
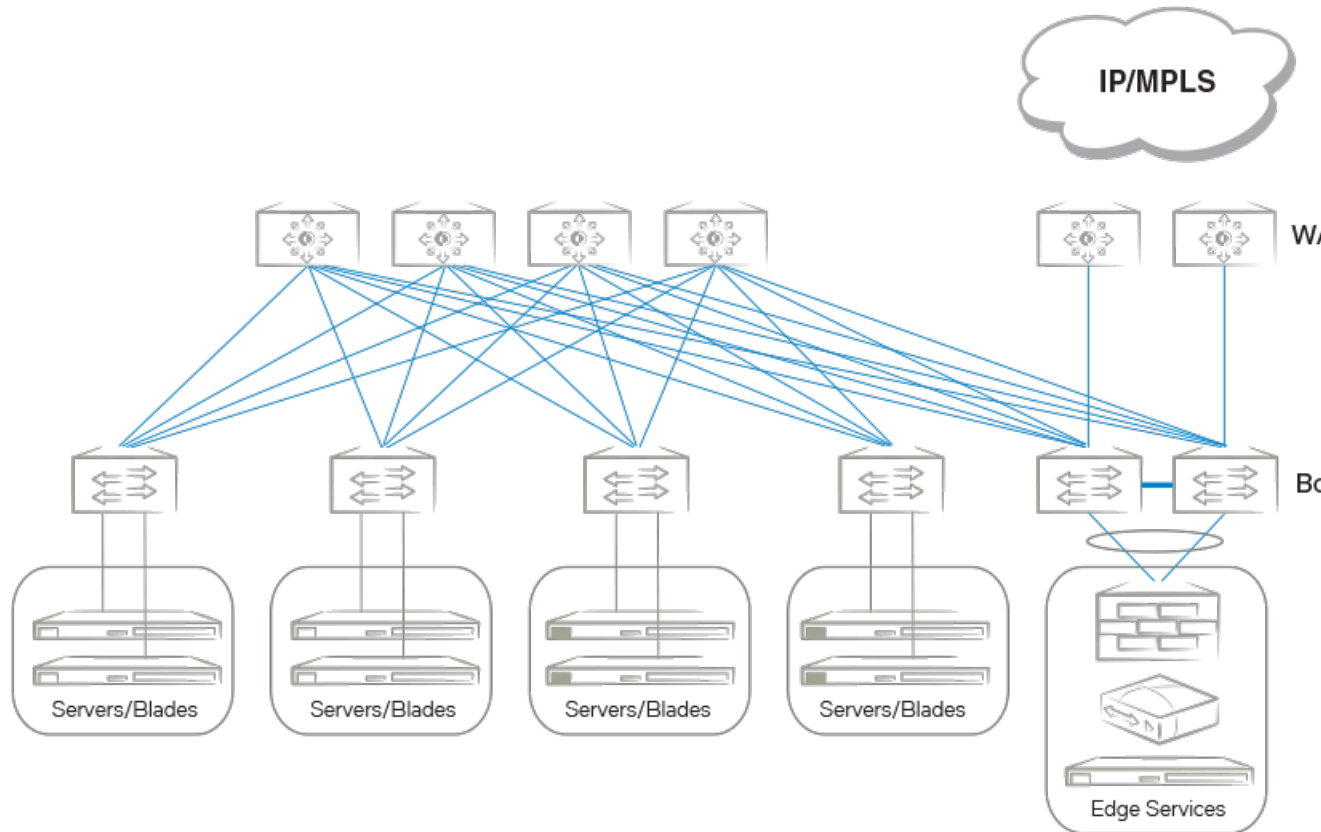
Figure 31: Border and service leaf

Colocated spine and border leaf

With a Layer 2/Layer 3 control-plane extension model, a border leaf may provide just a route forwarding service—which is merely spine functionality. In the absence of a separate border leaf, spine nodes can themselves act as route reflectors towards the WAN/core.

VXLAN overlay configuration

An overlay gateway configuration must be present and should remain in the active state in order to exchange routes in the VXLAN overlay network. An example configuration follows.

```
overlay-gateway gateway1
  type layer2-extension
  ip interface Loopback 1
  map vni auto
  activate
!

interface Loopback 1
  ip address 192.168.32.10/32
  no shutdown
!
```

> **Note**
> Either automatic or manual VLAN mapping is supported. Hybrid mode is not supported.

The following are criteria for accepting or originating routes in a VXLAN overlay network:

- An overlay gateway is configured.
- The overlay gateway is activated.
- The tunnel type is Layer 2 extension.
- The source VTEP IP address is configured. In case the source IP address is obtained from a loopback interface, the loopback interface must be up.
- VLAN/BD-to-VNI mapping (auto or manual) exists to accept and originate routes for a specific VLAN/BD.

The preceding criteria affect only the exchange of routes between BGP neighbors with VXLAN encapsulation. When the overlay gateway is deactivated or unconfigured, all dynamic tunnels are deleted, EVPN routes received with VXLAN encapsulation are removed, and the RIBout of the BGP EVPN neighbors configured with VXLAN encapsulation is flushed.

Dynamic VTEP discovery

Dynamic VTEP discovery is a configurable option. It can be enabled under BGP EVPN address-family configuration mode, as shown here.

> **Note**
> This feature is supported on static VXLAN tunnels with non-MCT nodes, for regular VTEPs but not for logical VTEPs.

```
router-bgp
  address-family l2vpn evpn
    vtep-discovery
!
```

By default VTEP discovery is enabled. When enabled, if a route with BGP encapsulation extended community as VXLAN type is imported into the MAC-VRF table, a VXLAN tunnel (if it does not already exist) is created in the system with the destination IP address as the next-hop IP address of the route. MAC learning in the forwarding plane on BGP-discovered VXLAN tunnels is automatically disabled.

When the preceding "vtep-discovery" is disabled, all dynamically discovered VXLAN tunnels are deleted from the system. VTEPs can be administratively configured, and they can be instructed to disable MAC learning in the forwarding plane and instead to use BGP, as in the following example.

```
overlay-gateway gateway1
  type layer2-extension
...
  site tunnel1
    ip address 192.168.32.20
    extend vlan add 10-100
    extend bridge-domain add 10-100
!
  activate
!
```

VLANs/BDs on static tunnels are not extended by BGP over inclusive-multicast routes received from remote nodes. They should be extended manually, as shown in the preceding configuration. Only when specific VLANs/BDs are extended does BGP download the corresponding EVPN routes pointing to the appropriate tunnel. A statically configured VXLAN tunnel overrides any dynamically discovered tunnel.

Similarly, when a static tunnel is removed, a dynamic VTEP is created if a route with that next-hop IP address is present in the BGP MAC-VRF table.

MCT and logical VTEP

MCT cluster configuration is required for logical VTEP (LVTEP) operation. MCT peers should have BGP EVPN route reachability with the corresponding encapsulation (MPLS). No manual BGP configuration is needed for MCT.

The following figure shows a typical example of a two-node MCT in a VXLAN IP Fabric. MCT cluster members have two BGP peerings: one with MCT cluster encapsulation (MPLS) and the other with VXLAN encapsulation to the spine.
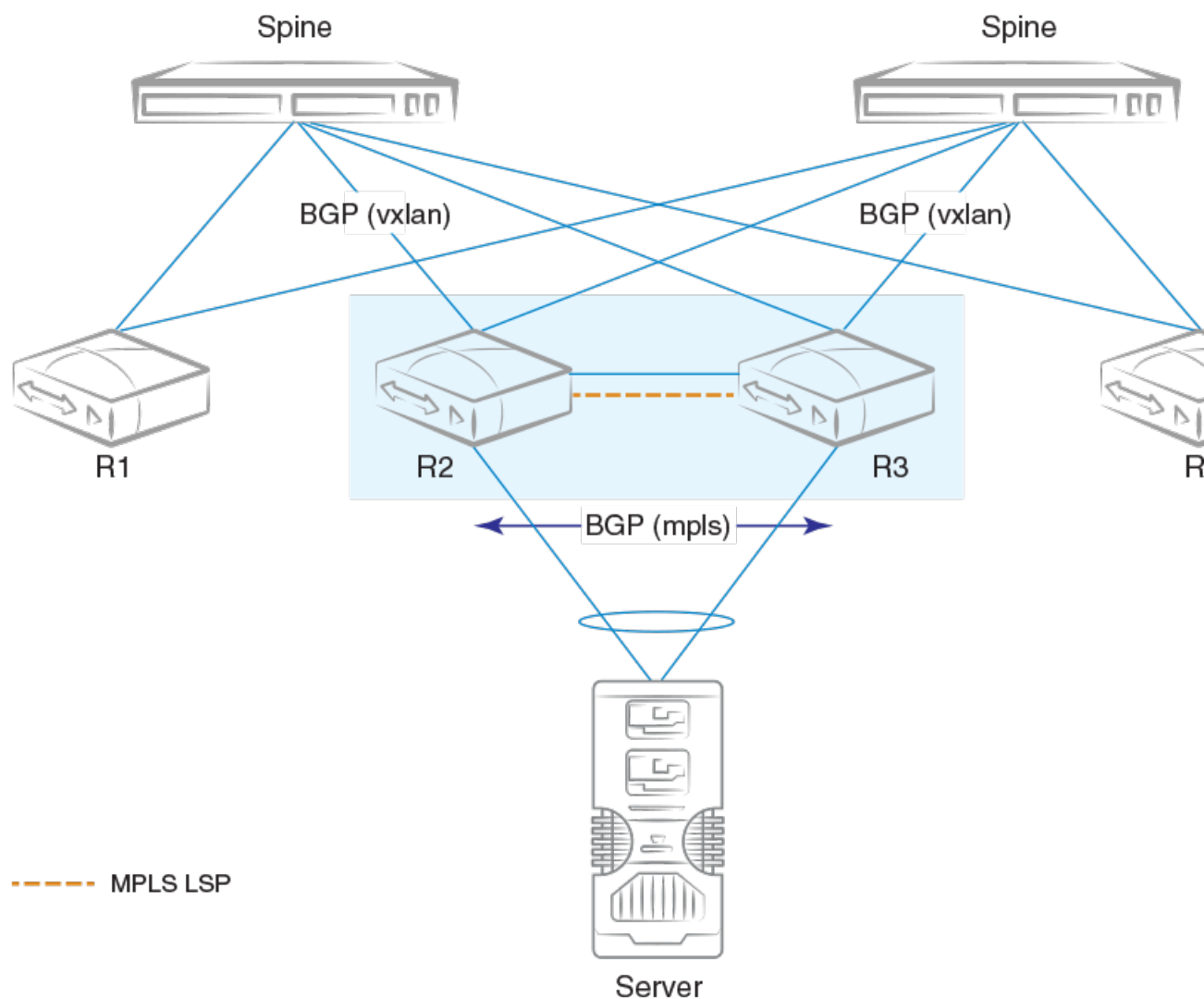
Figure 32: MCT with LVTEP and direct BGP peering

In the presence of an LVTEP, BGP with VXLAN encapsulation cannot be used to exchange EVPN routes between MCT peers, because (1) VXLAN routes are originated with a source VTEP IP address that is same across MCT peers, and (2) VXLAN as a transport between MCT peers is not supported.

The following are requirements or recommendations for overlay gateway configuration on MCT cluster nodes:

- Configuring MCT cluster nodes as individual VXLAN tunnel end points is not recommended. MCT cluster nodes should be configured to form an LVTEP if VXLAN overlay is required.
- The same source VTEP IP address should be configured on all nodes in an MCT cluster to form an LVTEP.
- The same set of VLANs/BDs should be configured in the system and added under EVPN on all nodes in an MCT cluster.
- VLAN/BD-to-VNI mapping should be the same on all nodes in an MCT cluster.
- If routing is enabled on a VLAN/BD, static anycast gateway is recommended.

EVPN next-hop resolution and tunnel-EVI membership

BGP EVPN next-hop is resolved according to the encapsulation type in BGP encapsulation extended community that is attached to the route.

EVPN routes are filtered if the encapsulation type in the route differs from the encapsulation type configured for the neighbor.

The following table summarizes the next-hop address set in the EVPN MP_REACH NLRI in BGP updates originated by a router irrespective of eBGP/iBGP peering type.

Table 32: Next-hop address for MPLS and VXLAN

MPLS	VXLAN
BGP peering IP address	Source VTEP IP address

When there are intermediate BGP routers between the service end points, and these routers do not participate in tunnel termination and reorigination without routing in the overlay, BGP neighbors on the intermediate routers should be configured to keep the next-hop unchanged. This requirement applies to all types of tunnels, including MPLS VXLAN.

Support for IPv6 overlay next-hops is not available. The next-hop value in the EVPN routes is "IPv4 address," irrespective of IPv4 or IPv6 BGP peering.

The edge/leaf router performs next-hop resolution according to the encapsulation type. In the case of VXLAN, if VTEP discovery is enabled, VXLAN tunnels are automatically created and destroyed. For MPLS, tunnels are discovered by means of LDP/RSVP protocols. BGP resolves the EVPN MPLS next-hop with the MPLS tunnel, and routes are installed only when the next-hop is reachable. In case the MPLS tunnel becomes unreachable, the EVPN next-hop is marked as unresolved and routes are removed from the system.

The inclusive multicast route is originated for each VLAN/BD configured under EVPN, and this route triggers the extension of a corresponding VLAN/BD on the tunnel at the remote end. Import rules for inclusive multicast routes are discussed in BGP routing tables on page 271. Inclusive-multicast routes are required to carry the P-Multicast Service Interface (PMSI) tunnel attribute (RFC 6514, Sec. 5) in the

path-attribute to signal the tunnel attributes. PMSI tunnel attribute fields are filled as shown in the following table.

Table 33: PMSI tunnel attribute fields and descriptions

Attribute Field	Description
Tunnel Type	Tunnel type is always "Ingress Replication" (value 6) for all types of tunnels. No other type of tunnel is supported currently.
MPLS label	This field carries the L2 VNI for VXLAN and the EVI label for MPLS.
Tunnel Identifier	This field carries the source VTEP IP address for VXLAN and the BGP peering IP address for MPLS.

In the case of LVTEP, both of the MCT cluster members advertise inclusive-multicast routes with different router IDs but the same next-hop. VLAN/BD membership of the tunnel is removed only when both advertising routers withdraw their inclusive-multicast routes. Therefore, consistent VLAN/BD configuration is required in an MCT cluster.

In the case of MPLS, there is always one tunnel between the MCT peers. In addition, inclusive-multicast route affects only the tunnel from which the route is originated. In order to perform load balancing of the unicast traffic, aliasing is used as discussed in the next section.

Layer 2 (MAC) route exchange

All dynamic MAC addresses learned on VLANs/BDs added to an EVPN configuration are exported to BGP EVPN automatically.

Routes are imported on the remote node according to the route-target match. The fields in the MAC routes are filled as shown in the following table.

Table 34: MAC route fields and descriptions

Field	Description
Route Distinguisher	Either the auto or manual RD value is used, depending on the VLAN/BD configuration under EVPN.
ESI	In case the MAC is learned on an MCT client interface, the ESI of the client interface is present. Otherwise this value is 0.
Ethernet Tag	This is value is 0.
MAC address	This is the static or dynamic MAC address learned locally.
IP address	For MAC route IP address, this value is absent.
MPLS Label1	This is the L2 VNI in case of VXLAN, and the EVI label for MPLS.
MPLS Label2	This is not present.

BGP MAC mobility extended community is attached to the route to carry a sticky flag and sequence number. Static MAC addresses configured on the system are advertised by BGP with a sticky flag. Routers importing a MAC route with a sticky flag install it as static and no MAC movement is allowed. In BGP best-path selection, a MAC route with a sticky flag is preferred over routes without a sticky flag, irrespective of the sequence number.

MAC address-table **show** output has been enhanced to show the remote VTEP IP address and route type as EVPN for a BGP-learned MAC over VXLAN, as in the following example.

```
device# show mac-address-table
Type Code - CL:Cluster Local MAC   CCL:Cluster Client Local MAC
            CR:Cluster Remote MAC  CCR:Cluster Client Remote MAC
VlanId/BDId   Mac-address      Type         State       Ports/LIF/PW/Tunnel
100 (V)       0000.0164.0100   Static       Inactive    Eth 0/31
101 (V)       0000.0164.0101   Static       Inactive    Eth 0/31
10 (V)        0000.0164.0010   Static       Active      Eth 0/31
10 (V)        0000.0191.0010   EVPN         Active       Tu 61441 (192.168.32.10)
10 (V)        0001.0191.0010   EVPN         Active       Tu 61441 (192.168.32.10)
10 (V)        0001.0221.0010   EVPN         Active       Tu 61441 (192.168.32.10)
```

MAC move detection and dampening

When a host moves from one port to another port on the same router, it is not considered a move. However, when the host moves from one router to another router, the MAC address undergoes a MAC move procedure. The router on which a MAC address is learned locally prefers the local address and advertises it to other routers in the network. If the MAC address is already present in BGP, the sequence number in the MAC mobility extended community is incremented in the route being advertised. A MAC route without MAC mobility extended community implicitly means sequence number 0. A router receiving a remote MAC route, with a sequence number greater than that of the locally advertised route, prefers the remote route and withdraws the local one.

MCT and LVTEP are two special cases in which a MAC move is not triggered. If the same MAC address is present in the BGP table and is learned from same next-hop or with same ESI, the MAC route is advertised with the same sequence number present in the existing route.

The frequent movement of a MAC address from one router to another router causes unnecessary churn in the network and is an indication of a malicious host or loop. When the number of moves for a given MAC address in a specified time (the default is 3 seconds) exceeds the specified number of moves (the default is 5), the MAC route is dampened. The EVPN MAC route dampening behavior differs from the BGP route flap dampening procedure specified in RFC 2439. When a MAC route is dampened, the local route is marked as best and is present in the forwarding tables. The best route selection based on sequence number is stopped until corrective action is taken. Default parameters of MAC route dampening can be modified by means of the following command under EVPN configuration mode:

```
device(config-evpn-default)# duplicate-mac-timer 5 max-count 3
```

Automatic restoration of dampened routes

According to BGP EVPN RFC 7432, once a MAC/IP route is dampened because of frequent moves, manual intervention is required to restore the route. Section 15.1 in the RFC states, "The PE MUST alert the operator and stop sending and processing any BGP MAC/IP Advertisement routes for that MAC address until a corrective action is taken by the operator."

A major drawback of this approach is that the route remains dampened and no processing of the updates is performed until the dampening state of the route is cleared manually. It may happen that after an initial frequent flap of the route, either one of the VMs goes away or the duplicate address situation is resolved. However, the route remains dampened until a network administrator intervenes. Automatic restoration of the route is desired in this case.

The following approach is taken to restore the dampened route.

When BGP detects that only one source of the route (that is, all NLRIs are received from same next-hop) has remained and the route is dampened, the route is added to the timer list, which is processed after 5 minutes. When the timer expires, if the second source of the route is seen again, the route is removed from the timer list and remains dampened. After the timer expires, the dampening state of the route is cleared and the route is restored after best-path selection.

Aliasing in MPLS

Aliasing, as described in RFC 7432 (Sec. 8.4), is specific to MPLS. Dual-homed MAC addresses are those that are advertised with a nonzero ESI value. The membership of an ES is advertised by the ES-AD route. When a MAC address is received with a nonzero ESI, instead of the next-hop of the route being used, the number of paths and respective VLAN/BD MPLS labels are inferred from AD-per-ES and AD-per-EVI routes, respectively.

The following figure shows a typical case of aliasing in MPLS. Router R3 receives MAC routes learned on MLAG with a nonzero ESI value only from R1. However, AD-per-ES routes for the same ESI value are received from both R1 and R2. In this case, R3 may form ECMP of two paths that lead to the same MCT cluster (ES) and load-balance the traffic towards the host through both R1 and R2.
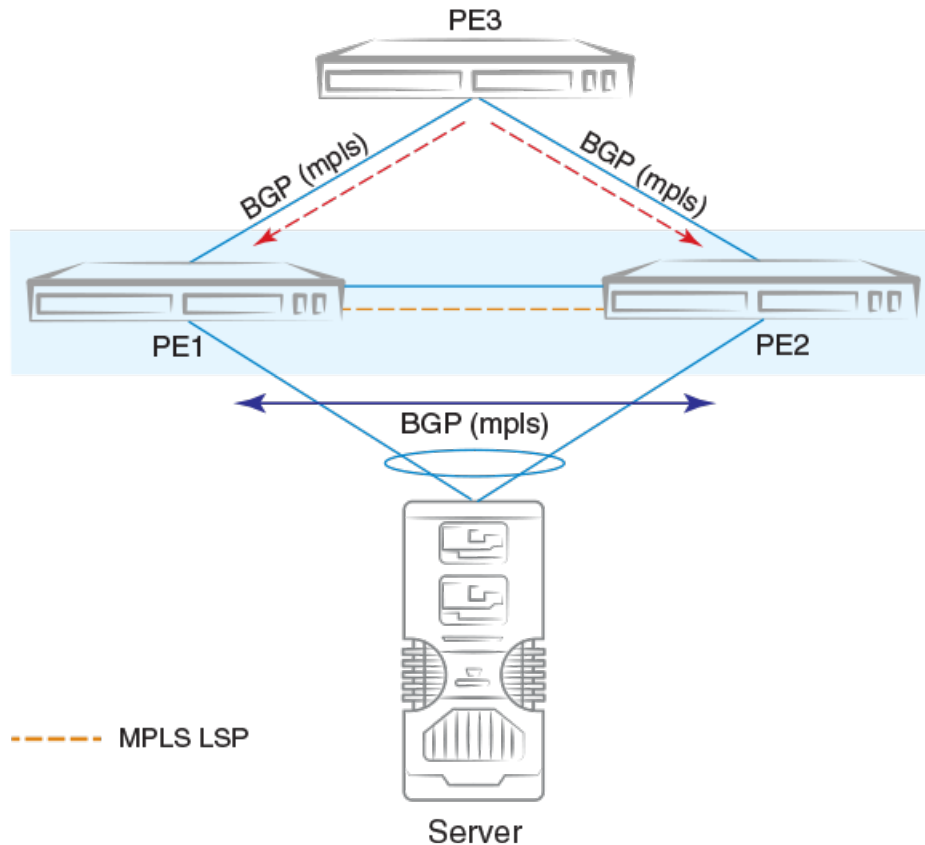
Figure 33: Aliasing in MPLS

Use case for AD-per-ES route

There are two use cases for AD-per-ES route on the non-MCT router:

- Faster withdrawal of MAC routes in case of ES link failures
- Load balancing of traffic for a given ES with AD-per-EVI routes

MAC/MACIP routes received with nonzero ESI value are not installed in the system unless at least one AD-per-ES route for corresponding ESI is present in BGP. Similarly, if MAC/MACIP routes are already installed and the last AD-per-ES route with corresponding ESI is withdrawn, routes are uninstalled from the forwarding plane.

Use case for AD-per-EVI route

AD-per-EVI routes are generated by MCT cluster members for each EVI configured on each ES. Because the number of AD-per-EVI routes can impose significant control-plane scaling issues, a maximum of 16 K routes is imposed.

The MPLS label field in the AD-per-EVI routes carries a per-EVI or per-ES-per-EVI MPLS label. Only per-EVI label allocation is assumed. With the assumption of a per-EVI label allocation scheme, it is not necessary to receive or originate AD-per-EVI routes for each ES or EVI. A subset of AD-per-EVI routes may suffice to identify the per-EVI label of the remote node.

BGP constructs a per-next-hop EVI-label map based on the AD-per-EVI routes. In addition, based on the reachability of AD-per-ES routes, an ECMP next-hop is constructed for Layer 2 routes.

Note that not all multihomed MAC addresses can attain forwarding plane ECMP behavior, because of limited forwarding resources. In case hardware resources are not available for ECMP in the forwarding plane, MAC addresses are sprayed across the available paths in software according to the VLAN hashing. Therefore, it is undeterministic to know whether stream-based load balancing for a given multihomed MAC address is available in the system.

Conversational MAC learning

Conversational MAC learning is not supported.

Example output for "show mac-address-table"

The following are example outputs of the **show mac-address-table** command.

```
device# show mac-address-table
Type Code - CL:Cluster Local MAC   CCL:Cluster Client Local MAC
            CR:Cluster Remote MAC  CCR:Cluster Client Remote MAC
VlanId/BDId   Mac-address      Type            State      Ports/LIF/PW/Tunnel
100 (V)       0000.0164.0100    Static          Inactive   Eth 0/31
101 (V)       0000.0164.0101    Static          Inactive   Eth 0/31
10 (V)        0000.0164.0010    Static          Active     Eth 0/31
10 (V)        0000.0191.0010    EVPN            Active     Tu 61441 (192.168.32.10)
10 (V)        0001.0191.0010    EVPN            Active     Tu 61441 (192.168.32.10)
10 (V)        0001.0221.0010    EVPN-Static  Active     Tu 61441 (192.168.32.10)
Total MAC addresses   :  6


device# show mac-address-table evpn
Type Code - CL:Cluster Local MAC   CCL:Cluster Client Local MAC
            CR:Cluster Remote MAC  CCR:Cluster Client Remote MAC
VlanId/BDId   Mac-address      Type            State      Ports/LIF/PW/Tunnel
10 (V)        0000.0191.0010    EVPN            Active     Tu 61441 (192.168.32.10)
10 (V)        0001.0191.0010    EVPN            Active     Tu 61441 (192.168.32.10)
10 (V)        0001.0221.0010    EVPN-Static  Active     Tu 61441 (192.168.32.10)
Total MAC addresses   :  3


device# show mac-address-table count evpn
EVPN Address Count: 50


device# show mac-address-table count
Dynamic Address Count: 2
Static Address Count: 0
Internal Address Count: 0
Local Address Count    : 2
Remote Address Count   : 0
EVPN Address Count: 2
Total MAC addresses    : 6


device# show mac-address-table count interface tunnel 61441
Dynamic Address Count: 2
Static Address Count: 0
EVPN Address Count: 2
Internal Address Count: 0
Local Address Count    : 2
```

```
Remote Address Count   : 0
Total MAC addresses    : 6


device# show mac-address-table interface tunnel 61441
Type Code - CL:Cluster Local MAC   CCL:Cluster Client Local MAC
          CR:Cluster Remote MAC  CCR:Cluster Client Remote MAC
VlanId/BDId   Mac-address      Type          State      Ports/LIF/PW
10 (V)        0000.0191.0010   EVPN-Static    Active     Tu 61441 (192.168.32.10)
10 (V)        0001.0191.0010   EVPN                      Active     Tu 61441
(192.168.32.10)
Total MAC addresses    :  2
```

ARP and ND (MACIP) Route Exchange

ARP and Neighbor Discovery (ND) addresses that are learned on the VLAN or Bridge Domain and added to EVPN are automatically exported into BGP. ARP and ND routes are referred to as MACIP (type 2) in RFC 7432.

The following types of ARP and ND routes are exported into BGP:

- Dynamically learned routes on VE interfaces with a local egress interface
- Statically configured routes on VE interfaces with a local egress interface
- IPv4 and IPv6 addresses configured on VE interfaces
- Virtual IPv4 and IPv6 addresses configured by means of VRRP or anycast gateway

If ARP or ND is learned with an egress interface as an MCT client interface, the corresponding ESI is attached to the MACIP route.

The resolution of ARP and ND received from BGP EVPN is based on the resolution of the corresponding MAC route. In addition, MAC route movement triggers the movement of MACIP routes. ARP and ND routes are withdrawn from BGP when the MAC resolution changes from a local egress interface to a tunnel.

Statically configured ARP and ND entries carry a sticky flag in the MAC mobility extended community and are installed as static ARP and ND on the routers importing this route. When an ARP or ND entry is sticky, the binding of the host address does not change on a local learning event.

Table 35: MACIP route fields and descriptions

Field	Description
Route Distinguisher	Either an auto or manual RD value is used, depending on the VLAN or bridge domain configuration under EVPN.
ESI	In case the MAC address is learned on an MCT client interface, the ESI of the client interface is present. Otherwise it is 0.
Ethernet Tag	This field is 0.
MAC address	This is the MAC address associated with the ARP or ND.
IP address	This is the IPv4 or IPv6 host address.

Table 35: MACIP route fields and descriptions (continued)

Field	Description
MPLS Label1	This is the L2 VNI in case of VXLAN and the EVI label for MPLS.
MPLS Label2	This is conditionally the VRF or Layer 3 VNI. If ARP or ND is learned on a non-default VRF and the VRF is configured to export routes into EVPN, the following behavior applies: • The MPLS Label2 field carries the Layer 3 VNI in case of VXLAN, and the VRF label in case of MPLS. Otherwise, Label2 is not sent in the route. • Both IP VRF RTs and VLAN/BD RTs are attached. VRRP and anycast gateway addresses are advertised with BGP default gateway extended community. They are used for logging errors when a MACIP route is imported but the same anycast IP address is not configured.

All EVPN MACIP routes are held in the ARP suppression cache, and this database is used for ARP suppression and conversational ARP.

ARP and ND suppression

When an ARP or ND binding for a host address is known in the control plane, the suppression feature allows a router to respond to the ARP or ND requests received on edge ports according to the control plane information rather than flooding the ARP and ND requests in the overlay network. This suppression prevents a significant amount of flooded traffic in the overlay network.

You can enable ARP and ND suppression on a VLAN or bridge domain. For more information, see ARP Suppression on page 35 and IPv6 Neighbor Discovery Suppression on page 62.

By default ARP and ND suppression are disabled.

> **Note**
> This behavior applies to both VXLAN and MPLS.

EVPN prefix route exchange and Multi-VRF support

Extensions to RFC 7432 to support the advertisement of IP prefix routes are proposed in IETF draft "IP Prefix Advertisement in EVPN."

A new route type, Type-5, is available for exchanging IPv4 and IPv6 prefixes in the overlay network. In a multitenant Layer 3 deployment, each tenant is assigned a VRF. Prefixes in each VRF may be gateway prefixes and prefixes belonging to the tenant, and these are discovered by routing protocols such as BGP and OSPF. In order to exchange prefixes in a VRF, end-to-end routing protocol adjacency in that VRF is required between the routers hosting a given tenant. EVPN prefix route eliminates such a limitation and, similar to IP VPN, can accumulate routes across all VRFs in BGP and allow the per-VRF import of prefixes by routers depending on the route-targets.

IP VPN is MPLS-only technology, whereas EVPN prefix route exchange provides the same functionality over any type of overlay encapsulation type supported by EVPN. Prefix route exchange is supported on the VXLAN and MPLS types of BGP EVPN encapsulation and peering.

For each tenant VRF from which prefixes are required to be exported into EVPN or imported from EVPN, the following configurations are required:

- Configure the VRF import/export route-targets for EVPN
- Select an integrated routing and bridging (IRB) interface in the VRF for routing.

The following example shows VRF RD and RT configurations in IPv4 and IPv6 address-families for the respective import and export of prefixes from and into EVPN.

```
vrf red
 rd 100:1
 evpn irb ve 10
 address-family ipv4 unicast
  route-target export 100:100 evpn
  route-target import 100:100 evpn
 !
 address-family ipv6 unicast
  route-target export 100:200 evpn
  route-target import 100:200 evpn
 !
!
```

The IRB interface is the VE interface that is used for routing after tunnel termination. The IRB interface should belong to the tenant VRF and be administratively up. It is not required to configure an IP address on the IRB interface, as shown in the following configuration example.

```
interface Ve 10
 vrf forwarding red
 no shutdown
 !
```

Depending on the encapsulation, appropriate attributes for overlay are used in the EVPN prefix route advertisement, as shown in the following table.

Table 36: Overlay attributes and descriptions

Field	Description
Route Distinguisher	This is configured under IP VRF
ESI	This is set to 0.
IP Prefix	This is the gateway or tenant prefix in the VRF.
Gateway IP	This is set to 0.
MPLS Label	VXLAN: This is the VNI corresponding to the IRB VE interface. MPLS: This is the IP VRF MPLS label

Export route-targets are attached to prefixes exported into EVPN. Remote end routers compare configured IP VRF RTs against RTs in the route and, in case of a match, prefix routes are imported into EVPN and then into IP VRF tables. In case RTs in a prefix route match multiple VRFs, routes from the EVPN table are imported into each matching VRF.

After the import into IP VRF, the next-hop resolution of the prefix route is not performed in the VRF. Instead, overlay next-hop is used and the prefix route is programmed with the exit interface as a tunnel.

In the case of VXLAN, the advertising router attaches the MAC address of the IRB interface as part of the MAC extended community of the BGP router in the prefix route. This MAC address is used as the next-hop MAC by the remote routers. In the forwarding plane, after tunnel decapsulation, routing on the inner frame is performed, because the outer DA MAC is the IRB interface MAC.

In the case of MPLS, MAC address of the router is not attached, because the inner VRF label in the frame causes routing in the corresponding VRF.

Symmetric or asymmetric routing

Depending on how VLANs/BDs are extended on the leaf nodes, symmetric or asymmetric routing may be observed. In the case of symmetric routing, routing on a VLAN/BD occurs only at a single router/leaf in the network. On the other hand, when the same VLAN/BD is extended on multiple routers, in the presence of distributed anycast gateway, each leaf node performs routing for traffic entering from local edge ports, causing routing to occur asymmetrically, depending on the direction of traffic.

Import/export route-map filtering

The route-map-based filtering of routes imported into IPVRF or exported into EVPN is supported. An import or export route-map can be applied under IPv4 or IPv6 address-family under a VRF configuration, as shown in the following example configuration.

```
device(vrf-red-ipv4-unicast)# import map my-import-filter evpn
device(vrf-red-ipv4-unicast)# export map my-export-filter evpn
```

Conversion of MACIP to host route to avoid traffic tromboning

In a typical IP Fabric network where distributed anycast gateway is configured on multiple leaf nodes, the same connected prefix is advertised by multiple nodes. A leaf node that does not extend the same VLAN/BD, but has extended the VRF, imports these prefixes, forms ECMP, and load balances the traffic across the tunnels. Because of ECMP, traffic may take a suboptimal path as shown in the following figure.
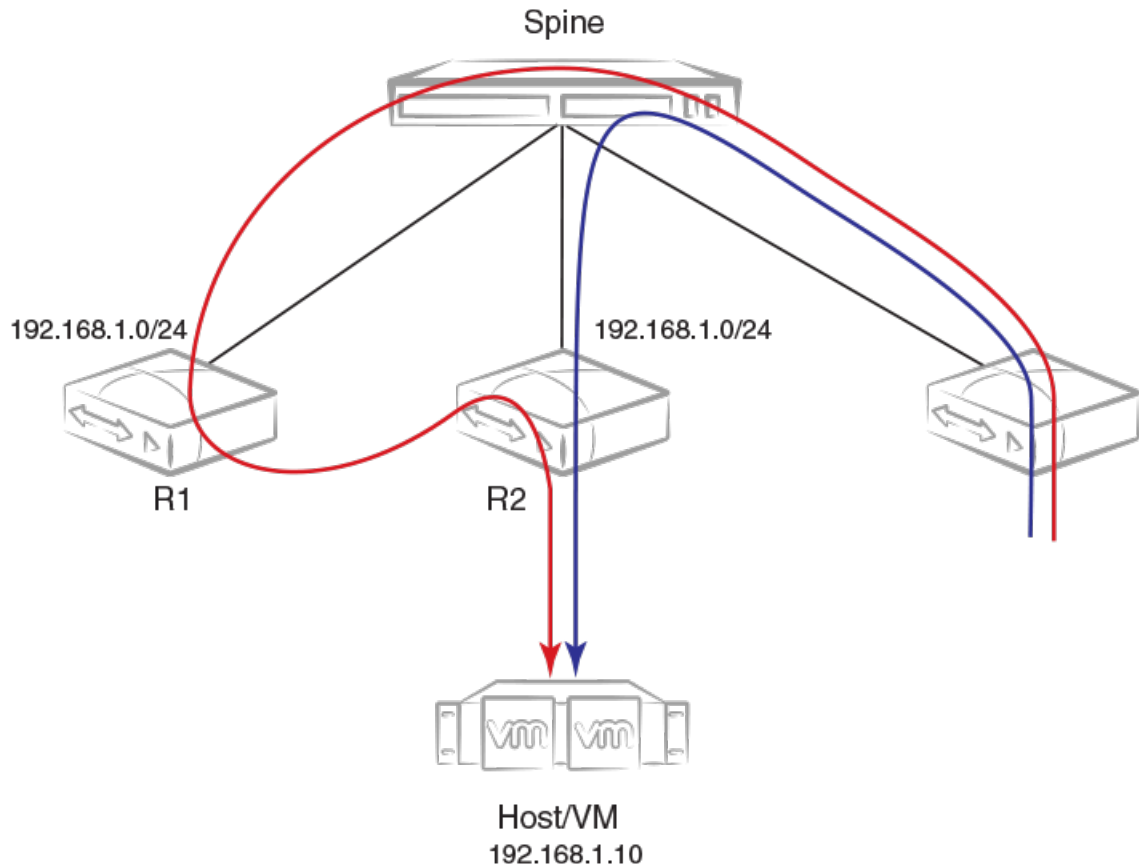
Figure 34: Avoiding suboptimal paths by converting MACIP routes to host routes

In order to avoid the suboptimal forwarding, a router extending the tenant VRF accepts MACIP routes matching the IP VRF RT in the EVPN table and converts those MACIP routes to IPv4 /32 prefix routes or IPv6 /128 or routes and imports them into the IP VRF table.

Anycast/VRRP IPv4 and IPv6 prefix routes are advertised with BGP default gateway extended community.

In the current release, because conversational ARP/ND is not supported, all converted IPv4 /32 host routes are installed in the forwarding plane. On the other hand, even though prefixes are received with default-gateway extended community, they are installed in the forwarding plane as normal prefix routes pointing to the ECMP next-hop.

By default, the conversion of MACIP routes to host routes is enabled. However, it can be disabled by means of the following configuration under BGP EVPN address-family.

Avoiding traffic tromboning in an MCT cluster

In an EVPN VXLAN network, Type-5 prefix routes are advertised by an MCT cluster with a source VTEP IP address that is common within the cluster. Also, each route carries a router-MAC address that is system-MAC unique to an MCT node. The remote node installing the prefix routes advertised by the MCT peer chooses one of the paths in the forwarding plane. However, an intermediate router may further load balance the traffic depending on the destination LVTEP IP address.

In the following figure, traffic towards Host/VM is load balanced by R3 across R1–R2 ECMP paths (the same next-hop but with different router-MAC addresses). However, the same traffic gets load balanced again by Spine across R1 and R2. Because of the underlay load balancing, Layer 3 traffic with the router-MAC of R2 may land on R1 and conversely. Because the destination MAC after VXLAN tunnel termination belongs to the MCT peer, traffic is Layer 2-switched to the MCT peer and is then routed there. This causes traffic tromboning.
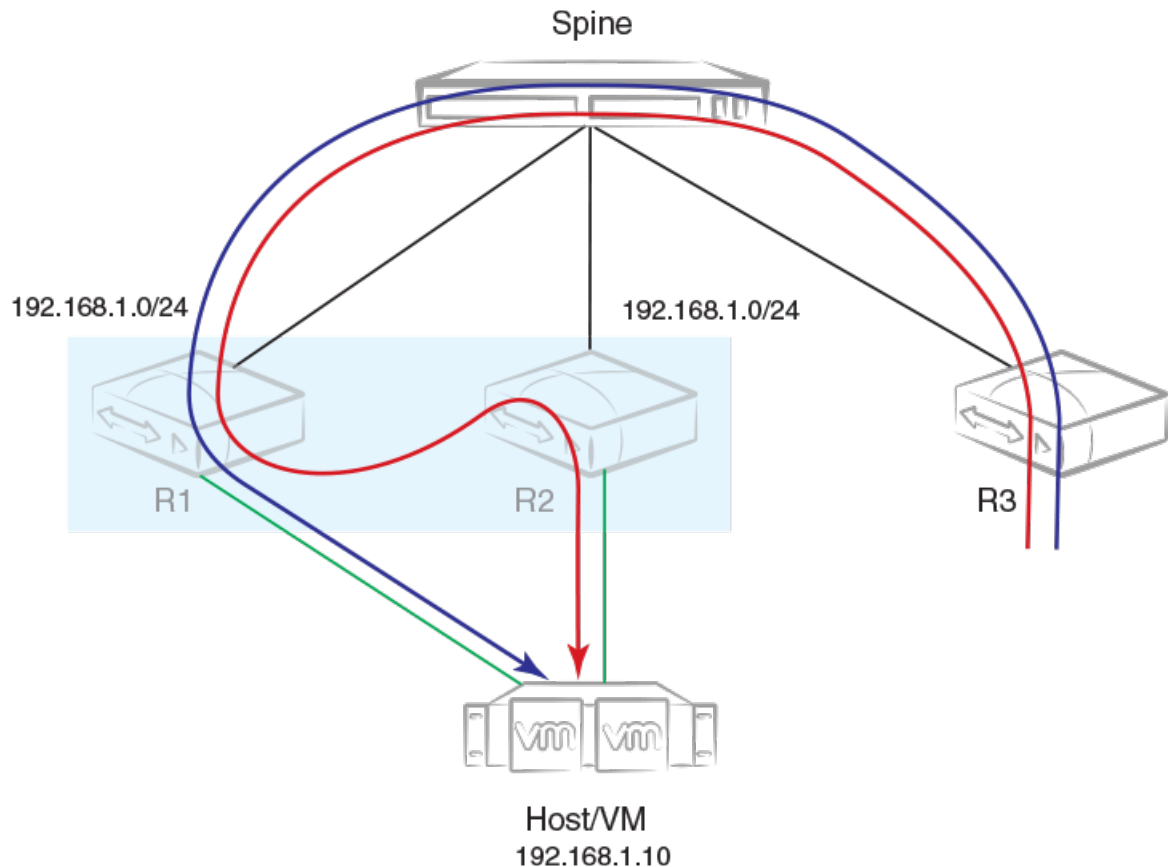
Figure 35: Avoiding traffic tromboning for EVPN Layer 3 traffic in an MCT cluster

This issue is resolved by exchanging the router-MAC address within the MCT cluster and installing the router MAC address of the MCT peer for routing based on the following example configuration.

When a peer gateway configuration is present for an EVPN IRB VE interface for the VRF, Layer 3 traffic destined for the MCT peer's router MAC in the corresponding VLAN/BD is routed locally.

BGP EVPN VXLAN data center interconnect

This section details the BGP EVPN VXLAN support for data center interconnect (DCI).

Data center Layer 2 interconnect between EVPN VXLAN and EVPN MPLS or VPLS is not supported. However, EVPN VXLAN and VPLS can coexist on the router, although a given VLAN/BD should not be extended into EVPN and be configured for VPLS.

The following sections address a variety of DCI interconnect scenarios.

Layer 2 and Layer 3 control-plane extension

In this scenario, VXLAN tunnels are extended by means of a border leaf instead of getting terminated. This extension is primarily the spine functionality being provided by the border leaf, except that the control and forwarding planes are extended over the WAN/core in the case of a border leaf.

Layer 2 handoff

In this scenario, VXLAN tunnels are terminated at the border leaf, as shown in the following figure. Depending on the interconnect technology being used between data centers extending over the WAN (such as VPLS, VXLAN Layer 2 extension, or any other interconnect scheme), the border leaf bridges the two domains through forwarding plane learning without any control plane extension.
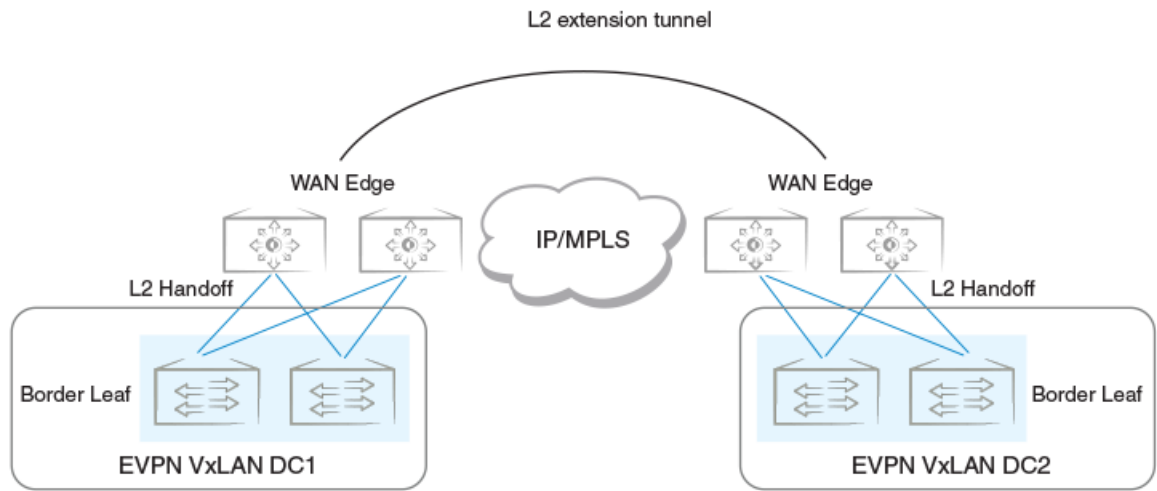
Figure 36: Layer 2 handoff-based DCI

Layer 3 handoff

In the case of Layer 3, all VRFs in the data center are terminated on the border leaf and traffic is routed towards the WAN, as shown in the following figure. Similarly, Layer 3 traffic received from the WAN is routed and forwarded over tunnels in the data center. Because Layer 3 routes across multiple VRFs can be imported into a single (interconnecting) VRF, all possible VRFs in a given data center do not have to be configured on the border leaf.
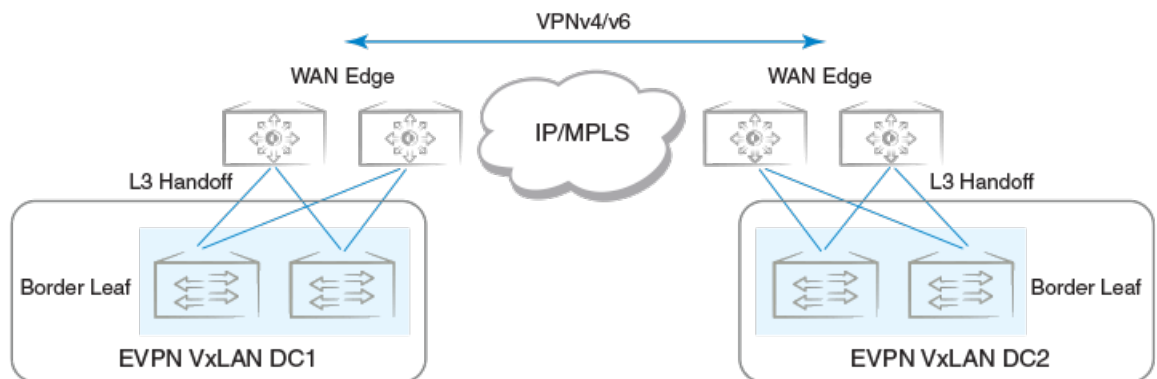
Figure 37: Layer 3 handoff-based DCI

Static anycast gateway

Static anycast gateway functionality provides support for seamless VM mobility across the leaf switches in IP Fabric deployments.

Hosts attached to the leaf switches are configured with a default gateway, which is typically the IP address of the leaf switch in the VLAN facing the host. If this host moves to another leaf switch, the host has to be reconfigured with the IP address of the new leaf switch to which it is now connected as its default gateway.

To make this movement of host from one leaf switch to another seamless, all the leaf switches are configured with the same anycast IP address and the associated virtual MAC (VMAC or anycast MAC) address for a given VLAN/BD. This configuration allows any leaf switch to behave as the default gateway for the host and allows for the most optimal forwarding behavior.

The ingress leaf switch recognizes the VMAC address as its own MAC address and does Layer 3 forwarding.

As shown in the following figure, for VLAN 10, all leaf switches are configured with the same MAC address and IP address.
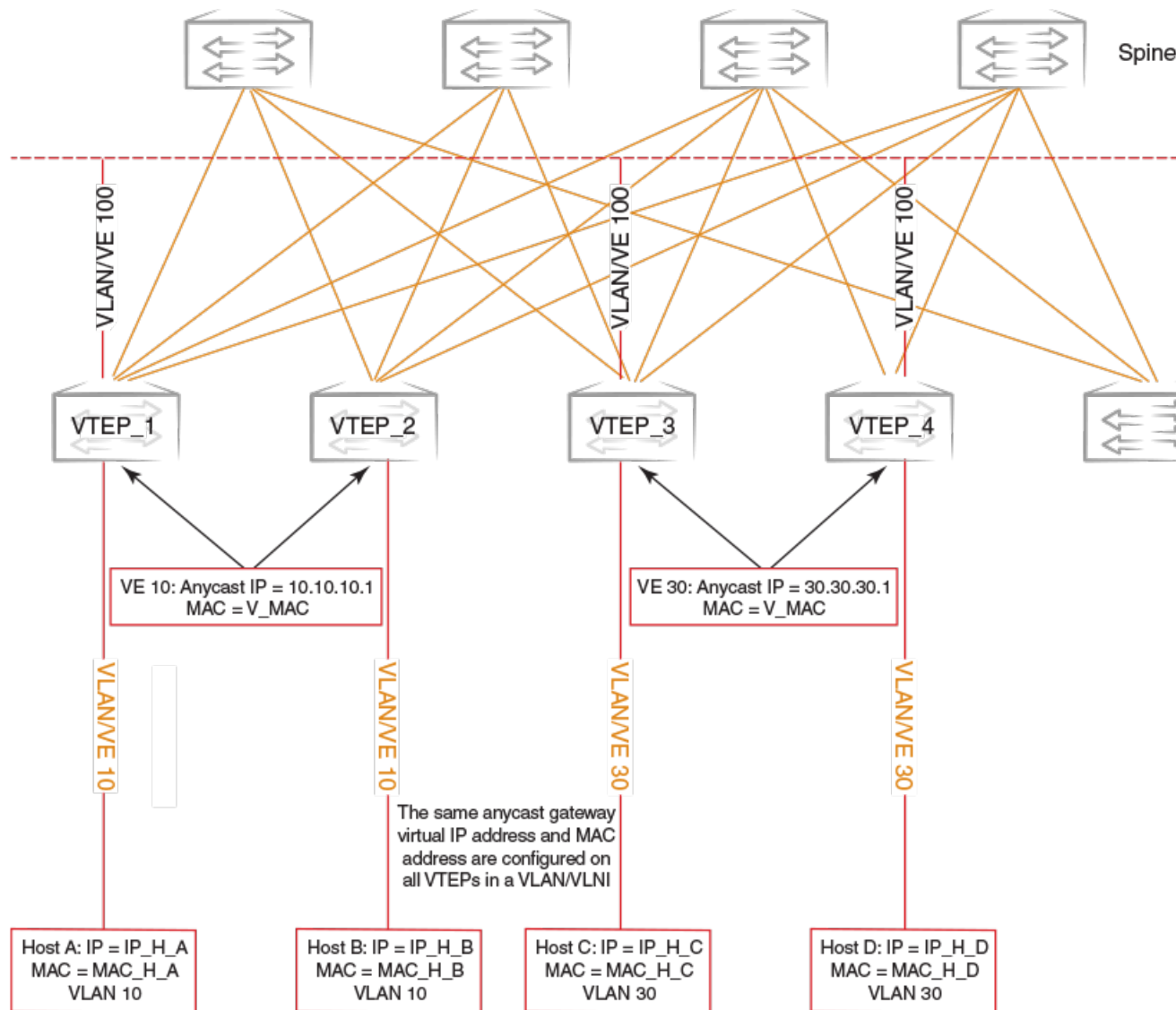
Figure 38: Static anycast gateway

When the host sends an ARP request for the gateway IP address on VLAN 10, the ingress leaf switch intercepts the ARP request and responds with the VMAC address associated with anycast IP. This behavior is controlled for each VE interface.

> **Note**
> Static anycast gateway is recommended only in the presence of a BGP EVPN control plane.

In order to configure static anycast gateway, configure the VMAC address first by specifying the default MAC address or an arbitrary unicast MAC address as in the following IPv4 and IPv6 examples. The default MAC address values are 02e0.5200.0100 and 02e0.5200.0200 for IPv4 and IPv6, respectively.

```
device(config)# ip anycast-gateway-mac default-mac
device(config)# ipv6 anycast-gateway-mac default-mac

device(config)# ip anycast-gateway-mac 0000.0101.0101
device(config)# ipv6 anycast-gateway-mac 0000.0101.0102
```

The anycast gateway IP address can be configured under the VE interface for VLANs/BDs, as in the following example.

```
device(config)# vlan 100
device(config-vlan-100)# router-interface ve 100
device(config-vlan-100)# int ve 100
device(config-if-Ve-100)# ip anycast-address 100.0.0.1/24
device(config-if-Ve-100)# ipv6 anycast-address 1000::1/24
```

> **Note**
> An IPv4 and IPv6 anycast MAC address cannot be configured as the same MAC address.

The configured anycast gateway address can be seen in the following examples.

```
device# show ip anycast-gateway
Gateway mac: 02e0.5200.0100
Ve10          1.1.1.0/24                              Inactive (Interface Down)
Ve100         100.0.0.1/24                            Active

device# show ipv6 anycast-gateway
Gateway mac: 02e0.5200.0200
Ve100         1000::1/24                              Active
```

> **Important**
> It is recommended that ARP suppression be configured on VLANs/BDs if static anycast gateway is configured on the corresponding VE interface. Otherwise, duplicate ARP responses to the gateway IP address are observed.

IP unnumbered interface

In a typical Layer 3 Clos network, routers are directly connected and require the assignment of IP addresses to each pair of routers, typically from the same subnet.

With large numbers of routers and redundant Layer 3 interfaces, a significant number of IP addresses are consumed just to configure the network itself. Using /31 masks reduces the consumption of addresses, but two IP addresses are still consumed per interface. Using unnumbered interfaces greatly reduces the number of IP addresses consumed in the configuration of the network.

This feature borrows an IP address from another Layer 3 interface already configured on the router. This address is used as a source address in the Layer 3 packets that are sent out the unnumbered interface. The interface from which the IP address is borrowed is called the donor interface. The following points highlight some of the key aspects of the use and functionality of this feature:

- Only physical interfaces can be configured as unnumbered interfaces. Unnumbered interfaces are not supported for virtual Ethenet (VE) or switched virtual interface (SVI) interfaces.
- The donor interface can be any other Layer 3 interface (Ethernet/VE/loopback).
- Unnumbered interface support is provided for IPv4 address family. Consequently, only an IPv4 address can be borrowed for an unnumbered interface.
- Once the interface is configured as an unnumbered interface, it is treated as a point-to-point interface and there can be only one remote neighbor attached to the interface.

- Because no network subnet is associated with the unnumbered interface, ARP is not supported on an unnumbered interface.
- Unnumbered interfaces are not supported in the management VRF, because routing protocols (OSPF/BGP) are not enabled in the management VRF.

The following diagram illustrates the overall concept of unnumbered interfaces and donor interfaces.
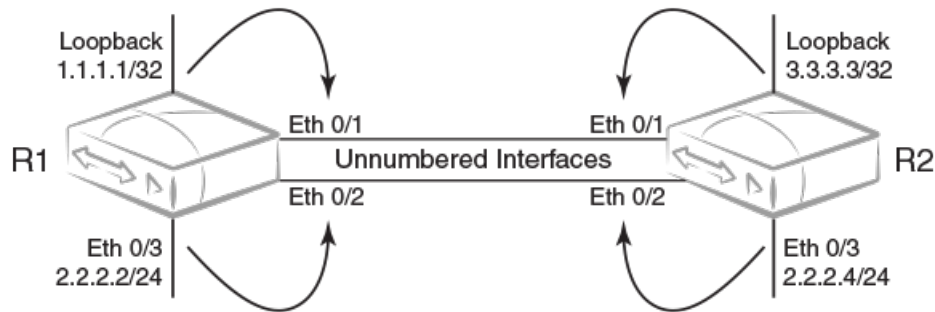
Figure 39: IP unnumbered and donor interfaces

- Interfaces Ethernet 0/1 and Ethernet 0/2 on R1 and R2 are unnumbered interfaces.
- Interfaces Ethernet 0/1 on R1 and R2 borrow IPv4 addresses from a loopback interface.
- Interfaces Ethernet 0/2 on R1 and R2 borrow IPv4 addresses from physical interface Ethernet 0/3.
- The end points of the unnumbered interface may or may not be in the same subnet.

The following functionalities are required to support Layer 3 over unnumbered interfaces:

- Neighbor discovery
- Host routes to reach neighbors over unnumbered interfaces
- Routes using unnumbered interfaces as next-hops

> **Note**
> For a simple three-stage IP Fabric, IP unnumbered interfaces can be used. For a five-stage IP Fabric, numbered interfaces are highly recommended, although IP unnumbered interfaces can be used in five-stage IP Fabric deployments if third-party devices are not included in the design. Extreme does not recommend using a mix of unnumbered and numbered interfaces within an IP Fabric.

High availability

BGP EVPN graceful restart (GR) capabilities are supported as follows:

- BGP EVPN GR helper is supported.
- BGP EVPN GR router restart is not supported.
- BGP Process-restart is not supported in BGP EVPN.

Generic Routing Encapsulation

GRE tunnels

The Generic Routing Encapsulation (GRE) is a transport protocol that transmits other protocol packets. The GRE tunnels act as a virtual point-to-point connection identified by the tunnel source and the tunnel destination configuration. A tunnel interface can be associated with an IP interface. A GRE tunnel acts as a transport medium for the IP interface. In SLX-OS, a VE interface is associated with a GRE tunnel. The properties such as the IP address, routing protocol that are configured in the VE interface are be applied for the GRE tunnel.

Before configuring a GRE tunnel, a point-to-point tunnel configuration, GRE requires both the ends of the tunnel to be configured.

The diagram below illustrates the GRE tunnel.

Figure 40: GRE tunnel
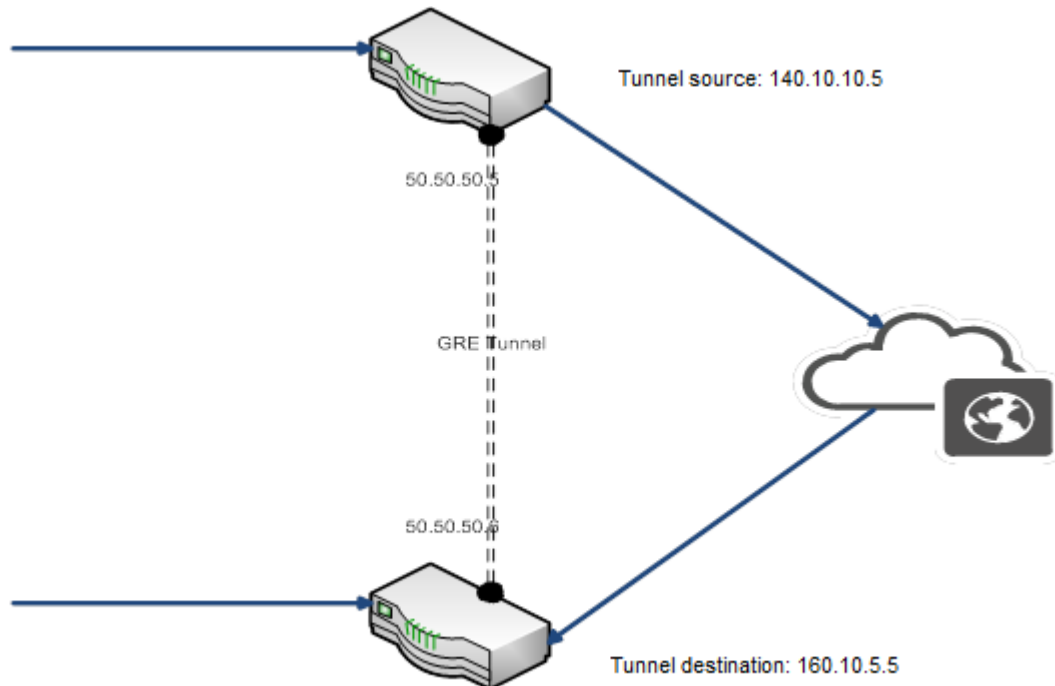
The IP address of a tunnel is derived from the VE interface. After a VE is associated with a tunnel, all the configuration under the VE such as IP address, routing protocol configurations are applied to the GRE tunnel interface. GRE tunnels support pipe and uniform QoS mode. By default, the QoS mode is set to pipe mode for all the tunnels.

By default, statistics for a tunnel is disabled. You can enable statistics for each tunnel by using the `statistics` command. Note that traffic loss might occur when enabling or disabling statistics on a tunnel. GRE keep alive is an optional configuration. The minimum granularity of the GRE keep alive time is 1 second.

You can configure MTU for each tunnel interface. The MTU configured in a VE is used for tunnels. When the VE MTU is set to the default MTU 1500, the GRE tunnel MTU will be adjusted to 1476. If the VE MTU is set to any value other than the default value, the VE MTU is used as the GRE tunnel MTU.

GRE Configuration Considerations

The GRE feature has some limitations and restrictions that you should understand before you configure it.

- The tunnel source and the destination addresses do not support VRF.
- GRE tunnels support only OSPF, ISIS, and BGP.
- QoS mutation configuration on the VE is not supported, including VEs that are bound to GRE tunnels.
- The maximum number of GRE tunnels supported is 256.
- The routing protocol used for resolving the GRE tunnel destination cannot be configured under the VE that is bound to the GRE tunnel.

- GRE tunnels do not support IPv6.
- GRE keep-alive is supported only with the default VRF. If you want GRE keep-alive in the non-default VRF, configure a static route in the VRF context to reach the GRE tunnel source.

Configuring a Tunnel

To configure a GRE tunnel, perform the following tasks.

Procedure

1. Enter global configuration mode.

```
device# configure terminal
```

2. Use the **interface tunnel** command to enter interface configuration mode and configure a tunnel.

```
device (config)# interface tunnel 5
```

The range is from 1 through 1024.

3. Use the **mode gre ip** command to configure GRE encapsulation over IP as mode.

```
device(config-intf-tunnel-5)# mode gre ip
```

4. Use the **source** command to configure the source IP address of the tunnel.

```
device(config-intf-tunnel-5)# source 10.1.1.10
```

The source IP address or interface that is configured will be used as the source of the GRE tunnel.

5. Use the **source** command to configure source interface of the tunnel.

```
device(config-intf-tunnel-5)# source ethernet ve 3
```

Ethernet, loopback, and VE interfaces are supported. The maximum number of tunnel source supported is 16.

> **Note**
> When the physical/ve interface is specified as the source of the GRE tunnel, the lowest IP address of that interface is used as the tunnel source IP address. If the smallest IP address is removed from the interface, the next smallest IP address is used as the tunnel source.

6. Use the **destination** command to configure destination IP address.

```
device(config-intf-tunnel-5)# destination 10.1.1.11
```

7. Use **router- interface** command to configure a router interface for the tunnel.

```
device(config-intf-tunnel-5)# router-interface ve 4
```

The tunnel source VE and the router interface VE cannot be the same.

8. Use the **no shutdown** command to configure the tunnel to remain up.

```
device(config-intf-tunnel-5)# no shutdown
```

Example

The following example configures a GRE tunnel.

```
device# configure terminal
device (config)# interface ve 3
device(config-if-Ve-3)# ip address 50.50.50.5/24
device(config-if-Ve-3)# exit
```

```
device (config)# interface tunnel 5
device(config-intf-tunnel-5)# mode gre ip
device(config-intf-tunnel-5)# source 10.1.1.10
device(config-intf-tunnel-5)# source ve 4
device(config-intf-tunnel-5)# destination 10.1.1.11
device(config-intf-tunnel-5)# router-interface ve 3
device(config-intf-tunnel-5)# dscp-ttl-mode pipe
device(config-intf-tunnel-5)# no shutdown
```

Binding a tunnel to a VE interface

Procedure

1. Enter global configuration mode.

   ```
   device# configure terminal
   ```

2. Use the **interface** command to enter interface configuration mode and configure a VE interface.

   ```
   device(config)# interface ve 3
   ```

   The range is from 1 through 4096.

3. Use the **ip address** command to assign an IP address to the VE interface.

   ```
   device(config-if-Ve-3)# ip address 10.1.1.1
   ```

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

   ```
   device(config-if-Ve-3)# exit
   ```

5. Use the **interface** command to enter interface configuration mode and configure a tunnel.

   ```
   device(config)# interface tunnel 5
   ```

   The range is from 1 through 1024.

6. Use the **router-interface** command to configure a router interface for the tunnel.

   ```
   device(config-intf-tunnel-5)# router-interface ve 3
   ```

Example

This example shows how to bind a tunnel to a VE interface.

```
device# configure terminal
device (config)# interface ve 3
device(config-if-Ve-3)# ip address 10.1.1.1
device(config-if-Ve-3)# exit
device (config)# interface tunnel 5
device(config-intf-tunnel-5)# router-interface ve 3
```

Enabling tunnel statistics

Procedure

1. Enter global configuration mode.

   ```
   device# configure terminal
   ```

2. Use the **interface** command to enter interface configuration mode and configure a tunnel.

   ```
   device(config)# interface tunnel 5
   ```

   The range is from 1 through 1024.

3. Use the **statistics** command to enable statistics on the tunnel.

```
device(config-intf-tunnel-5)# statistics
```

Example

The following example shows how to enable tunnel statistics.

> **Note**
> GRE tunnel statistics only counts traffic entering the GRE tunnel. Traffic egressing the GRE tunnel is not counted.

```
device# configure terminal
device (config)# interface tunnel 5
device(config-intf-tunnel-5)# statistics
```

Configuring GRE tunnel keepalive

Procedure

1. Enter global configuration mode.

```
device# configure terminal
```

2. Enter interface configuration mode and configure a tunnel.

```
device(config)# interface tunnel 5
```

Valid values range from from 1-1024.

3. Configure the keepalive interval and number of retries.

```
device(config-intf-tunnel-5)# keepalive time-interval 30 retry-count 2
```

Valid values for the interval range from 1-32767. Valid values for retries range from 1-255.

Configuring DSCP for the GRE tunnel

Procedure

1. Enter global configuration mode.

```
device# configure terminal
```

2. Enter interface configuration mode and configure a tunnel.

```
device(config)# interface tunnel 5
```

The range is from 1 thorough 1024.

3. Configure tunnel differentiated services codepoint (DSCP).

```
device(config-intf-tunnel-5)# dscp 10
```

Valid values range from from 0-63.

Configure MTU

Procedure

1. Enter global configuration mode.

```
device# configure terminal
```

2. Enter interface configuration mode and configure a VE interface.

```
device (config)# interface ve 5
```

The range is from 1 through 4096.

3. Configure MTU on the interface.

```
device(config-intf-ve-5)# ip mtu 9011
```

The range is from 1300 through 9194. The default MTU is 1476.

> **Note**
> Only the default MTU of 1476 is supported for GRE tunnels. IP packets up to 1476 bytes can be transmitted on the GRE tunnels. As a best practice, set the MTU of the traffic source interface to 1476, so that no fragmentation is needed.

Example

This example summarizes the commands in this procedure.

```
device# configure terminal
device(config)# interface ve 5
device(config-intf-ve-5)# ip mtu 9011
```

Displaying tunnel statistics

Procedure

1. From user EXEC mode, use the **show tunnel statistics** command to display statistics of all tunnels.

```
device# show tunnel statistics
```

2. Enter the tunnel ID to display statistics of a specific tunnel.

```
device# show tunnel statistics 7
```

Example

The following example shows how to display statistics of all tunnels.

```
device# show tunnel statistics
```

Example

The following example shows how to display statistics of the specified tunnel ID.

```
device# show tunnel statistics mode gre
Tnl ID   RX packets       TX packets       RX bytes         TX bytes
======== ================ ================ ================ =================
5        0                0                (NA)             0
```

Example

The following example shows how to display tunnel statistics mode.

```
device# show tunnel statistics mode gre
Tnl ID   RX packets       TX packets       RX bytes         TX bytes
======== ================ ================ ================ =================
10       0                10               (NA)             640
```

```
11       0              20             (NA)           1280
12       0              50             (NA)           22000
```

Example

The following example shows how to display information of the specified tunnel.

```
device# show tunnel 10
Tunnel 10, mode GRE
Ifindex 0x7c40000a, Admin state up, Oper state up
Source IP 14.101.0.4, Vrf default-vrf
Destination IP 15.10.0.3
Tunnel IP Interface : Ve 501 up
Tunnel TTL 255      Tunnel DSCP 0
Tunnel QosMode PIPE
Keepalive Interval 10000  RetryCount 3 TimeRemaining 27861 msecs
GRE Keep Alive : RX 62       TX 62

Active next hops:
    IP: 13.10.0.3, Vrf: default-vrf
    Egress L3 port: Ve 10, Outer SMAC: 609c.9f0d.4a14
    Outer DMAC: 001b.ed9f.1700
    Egress L2 Port: Unknown, Outer ctag: 0, stag:0, Egress mode: Local
    BUM forwarder: no
```

Clearing tunnel statistics

Procedure

1. From Privileged EXEC mode, use the **clear tunnel statistics** command to clear statistics of all tunnels.

   ```
   device# clear tunnel statistics
   ```

2. Enter the tunnel ID to clear statistics of a specific tunnel.

   ```
   device# clear tunnel statistics 7
   ```

Example

The following example shows how to clear statistics of all tunnels.

```
device# clear tunnel statistics
```

Example

The following example shows how to clear statistics of the specified tunnel.

```
device# clear tunnel statistics 7
```

IS-IS (IPv4)

IS-IS overview

The Intermediate System to Intermediate System (IS-IS) protocol is a link-state Interior Gateway Protocol (IGP) that is based on the International Standard for Organization/ International Electrotechnical Commission (ISO/IEC) Open Systems Internet Networking model (OSI). In IS-IS, an intermediate system (router) is designated as either a Level 1 or Level 2 device. A Level 1 router routes traffic only within the area in which the router resides. A Level 2 router routes traffic between areas within a routing domain.

The implementation of IS-IS is based on the following specifications and draft specifications:

- ISO/IEC 10589 - "Information Technology - Telecommunication and information exchange between systems - Intermediate system to Intermediate system intra-domain routing information exchange protocol for use in conjunction with the protocol for providing the connection less-mode Network Service (ISO 8473)", 1992

- ISO/IEC 8473 - "Information processing systems - Data Communications - Protocols for providing the connectionless-mode network service", 1988

- ISO/IEC 9542 - "Information Technology - Telecommunication and information exchange between systems - End system to Intermediate system intra-domain routing information exchange protocol for use in conjunction with the protocol for providing the connection less-mode Network Service (ISO 8473)", 1988

- RFC 1195 - "Use of OSI IS-IS for Routing in TCP/IP and Dual Environments", 1990.

- RFC 2763 - "Dynamic Host Name Exchange Mechanism for IS-IS", 2000.

- RFC 2966 - "Domain-wide Prefix Distribution with Two-Level IS-IS", 2000

- RFC 3373 - "Three-Way Handshake for Intermediate System to Intermediate System (IS-IS) Point-to-Point Adjacencies", 2002

- Portions of the Internet Draft "IS-IS extensions for Traffic Engineering" draft-ieff-isis-traffic-02.txt (dated 2000). that describe the Extended IP reachability type-length-value (TLV type 135) and the extended Intermediate System (IS) reachability TLV (TLV type 22). These portions provide support for the wide metric version of IS-IS. No other portion is supported on Extreme's implementation of IS-IS.

> **Note**
> The Extreme device does not support routing of Connectionless-Mode Network Protocol (CLNP) packets. The Extreme device uses IS-IS for TCP/IP only.

Relationship to the IP route table

The IS-IS routes are calculated and first placed in the IS-IS route table. The routes are then transferred to the IP route table.

The best IS-IS path for a given destination is sent to the IP route table for comparison to the best paths from other protocols to the same destination. The CPU selects the path with the lowest administrative distance and places that path in the IP route table:

- If the path provided by IS-IS has the lowest administrative distance, then the CPU places that IS-IS path in the IP route table.
- If a path to the same destination supplied by another protocol has a lower administrative distance, the CPU installs the other protocol's path in the IP route table instead.

The **administrative distance** is a protocol-independent value from 1 - 255. Each path sent to the CPU, regardless of the source of the path (IS-IS, OSPF, static IP route, and so on) has an administrative distance.

Each route source has a default administrative distance. The default administrative distance for IS-IS is 115.

You can change the administrative distance for IS-IS and other routes sources.

Intermediate systems and end systems

IS-IS uses the following categories to describe devices within an IS-IS routing domain (similar to an OSPF Autonomous System):

- Intermediate System (IS) - A device capable of forwarding packets from one device to another within the domain. In Internet Protocol (IP) terminology, an IS is a router.
- End System (ES) - A device capable of generating or receiving packets within the domain. In IP terminology, an ES is an end node or IP host.

When you configure IS-IS on a device, the device is an IS.

Figure 41 shows an example of an IS-IS network.
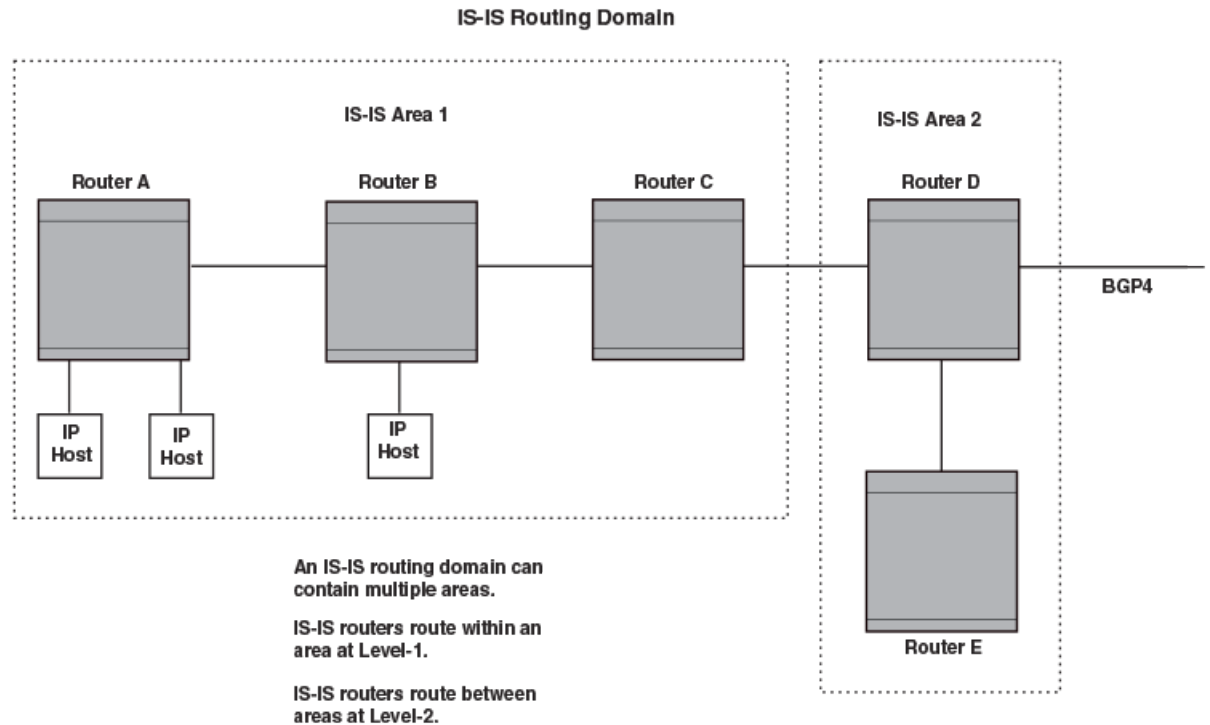
Figure 41: An IS-IS network contains Intermediate Systems (ISs) and host systems

> **Note**
> Since the implementation of IS-IS does not route OSI traffic but instead routes IP traffic, IP hosts are shown instead of ESs.

The other basic IS-IS concepts illustrated in this figure are explained in the following sections.

Domain and areas

IS-IS is an IGP, and thus applies only to routes within a single routing domain. However, you can configure multiple areas within a domain. A device can be a member of one area for each Network Entity Title (NET) you configure on the device. The NET contains the area ID for the area the NET is in.

In Intermediate systems and end systems on page 307, Routers A, B, and C are in area 1. Routers D and E are in area 2. All the routers are in the same domain.

Level-1 routing and Level-2 routing

You can configure an IS-IS router to perform one or both of the following levels of IS-IS routing:

> **Note**
> The ISO/IEC specifications use the spelling "routeing", but this document uses the spelling "routing" to remain consistent with other Extreme documentation.

- Level-1 - A Level-1 router routes traffic only within the area the router is in. To forward traffic to another area, the Level-1 router sends the traffic to its nearest Level-2 router.

- Level-2 - A Level-2 router routes traffic between areas within a domain.

In Intermediate systems and end systems on page 307, Routers A and B are Level-1s only. Routers C and D are Level-1 and Level-2 ISs. Router E is a Level-1 IS only.

Neighbors and adjacencies

A device configured for IS-IS forms an adjacency with each of the IS-IS devices to which it is directly connected. An adjacency is a two-way direct link (a link without router hops) over which the two devices can exchange IS-IS routes and other protocol-related information. The link is sometimes called a "circuit". The devices with which the device forms adjacencies are its neighbors , which are other ISs.

In Intermediate systems and end systems on page 307, Router A has an IS-IS adjacency with Router B. Likewise, Router B has an IS-IS adjacency with Router A and Router C.

Designated IS

A Designated IS is an IS-IS router that is responsible for gathering and distributing link state information to other Level 1 or Level 2 ISs within the same broadcast network (LAN). The Level 1 and Level 2 Designated ISs within a broadcast network are independent, although the same device can be a Level 1 Designated IS and a Level 2 Designated IS at the same time.

The Designated IS is elected based on the priority of each IS in the broadcast network. When an IS becomes operational, it sends a Level 1 or Level 2 Hello PDU to advertise itself to other ISs. If the IS is configured to be both a Level 1 and a Level 2 IS, the IS sends a separate advertisement for each level:

- The Level 1 IS that has the highest priority becomes the Level 1 Designated IS for the broadcast network.
- The Level 2 IS that has the highest priority becomes the Level 2 Designated IS for the broadcast network.

If the Designated IS becomes unavailable, for example in the case of a reboot, the IS with the next highest priority becomes the new IS. If two or more ISs have the highest priority, the IS with the highest MAC address becomes the Designated IS.

The priority is an interface parameter. Each interface that is enabled for IS-IS can have a different priority.

The figure below shows an example of the results of Designated IS elections. For simplicity, this example shows four of the five routers in Intermediate systems and end systems on page 307, with the same domain and areas.
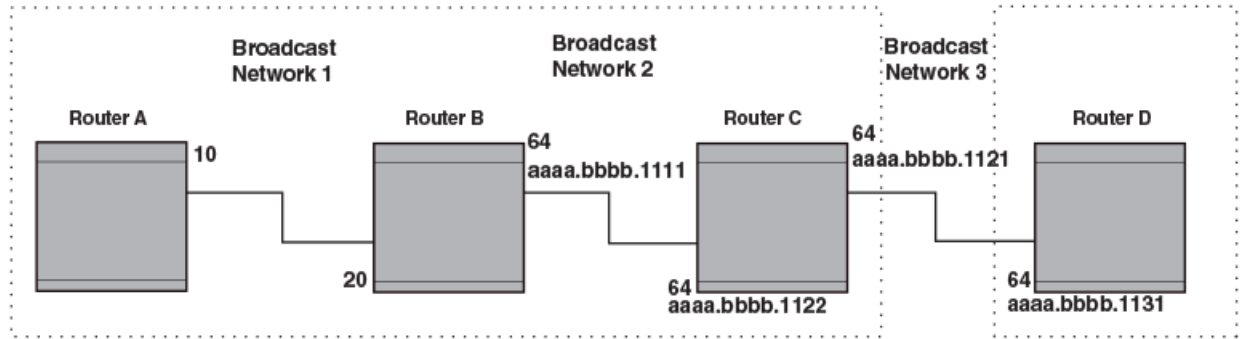
Figure 42: Each broadcast network has a Level-1 Designated IS and a Level-2 Designated IS

Designated IS election has the following results in this network topology:

- Router B is the Level 1 Designated IS for broadcast network 1
- Router C is the Level 1 Designated IS for broadcast network 2
- Router D is the Level 2 Designated IS for broadcast network 3

In this example, the IS-IS priorities for the IS-IS interfaces in broadcast network 1 have been changed by an administrator. The priorities for the interfaces in the other broadcast networks are still set to the default (64). When there is a tie, IS-IS selects the interface with the highest MAC address.

Broadcast pseudonode

In a broadcast network, the Designated IS maintains and distributes link state information to other ISs by maintaining a pseudonode . A pseudonode is a logical host representing all the Level 1 or Level 2 links among the ISs in a broadcast network. Level 1 and Level 2 have separate pseudonodes, although the same device can be the pseudonode for Level 1 and Level 2.

Route calculation and selection

About This Task

The Designated IS uses a Shortest Path First (SPF) algorithm to calculate paths to destination ISs and ESs. The SPF algorithm uses Link State PDUs (LSPDUs) received from other ISs as input, and creates the paths as output.

After calculating the paths, the Designated IS then selects the best paths and places them in the IS-IS route table. The Designated IS uses the following process to select the best paths.

Procedure

1. Prefer the Level 1 path over the Level 2 path.
2. If there is no Level 1 path, prefer the internal Level 2 path over the external Level 2 path.
3. If there is still more than one path, prefer the path with the lowest metric.
4. If there is more than one path with the lowest metric, load share among the paths.

   After selecting the best path to a destination, the software places the path in the IS-IS route table.

Three-way handshake for point-to-point adjacencies

Three-Way Handshake for Point-to-Point adjacencies provides three-way handshake mechanisms on point-to-point interfaces for the following benefits:

- Identifies neighbor restarts within the holding time period
- Identifies uni-directional link failures and stops forming of an adjacency with a peer where such link failures occur.

IS-IS CLI levels

The CLI includes various levels of commands for IS-IS. The figure below illustrates these levels.

Figure 43: IS-IS CLI levels

The following IS-IS CLI configuration modes are available

- ISIS router configuration mode: A global level for the configuration of the IS-IS protocol. At this level, all IS-IS configurations apply to IPv4 and IPv6.
- ISIS address-family IPv4 unicast configuration mode: Commands entered in ISIS address-family IPv4 unicast configuration mode apply only to IS-IS IPv4 configurations.
- ISIS address-family IPv6 unicast configuration mode: Commands entered in ISIS address-family IPv6 unicast configuration mode apply only to IS-IS IPv6 configurations.
- Interface subtype configuration mode: Various IS-IS options are configured on a specified interface.

Refer to the relevant Command Reference for more information on each of these configuration modes and for information on all of the commands accessed in these configuration modes.

Each address family configuration level allows you to access commands that apply to that particular address family only. To enable a feature in a particular address family, you must specify any associated commands for that feature in that particular address family. You cannot expect the feature, which you may have configured in the IPv4 IS-IS unicast address family, to work in the IPv6 IS-IS unicast address family unless it is explicitly configured in the IPv6 IS-IS unicast address family.

Enabling IS-IS globally

When IS-IS is enabled on a device, the device enters IS-IS router configuration mode. Several commands can then be accessed that allow the configuration of IS-IS.

About This Task

Procedure

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

   ```
   device# configure terminal
   ```

2. Enter the **router isis** command to enter IS-IS router configuration mode and enable IS-IS on the device.

   ```
   device(config)# router isis
   ISIS: Please configure NET!
   ```

3. Enter the **net** command and specify a NSAP address to configure a NET for IS-IS.

   ```
   device(config-isis-router)# net 49.2211.0000.00bb.cccc.00
   ```

Example

The following example enables IS-IS on a device.

```
device# configure terminal
device(config)# router isis
ISIS: Please configure NET!
device(config-isis-router)# net 49.2211.0000.00bb.cccc.00
```

Configuring the IS-IS IPv4 unicast address family

The IS-IS IPv4 unicast address family allows you to configure IPv4 IS-IS unicast settings that are separate and distinct from IPv6 IS-IS unicast settings. The following task enables ISIS IPv4 address family unicast configuration mode where a variety of IS-IS features can be configured.

About This Task

Procedure

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

   ```
   device# configure terminal
   ```

2. Enter the **router isis** command to enter IS-IS router configuration mode and enable IS-IS on the device.

   ```
   device(config)# router isis
   ```

3. Enter the **address-family ipv4 unicast** command to enable ISIS address-family IPv4 unicast configuration mode.

   ```
   device(config-isis-router)# address-family ipv4 unicast
   ```

Example

The following example enables ISIS address-family IPv4 unicast configuration mode.

```
device# configure terminal
device(config)# router isis
```

```
device(config-isis-router)# address-family ipv4 unicast
device(config-router-isis-ipv4u)#
```

Overload bit

If an IS's resources are overloaded and are preventing the IS from properly performing IS-IS routing, the IS can inform other ISs of this condition by setting the overload bit in LSPDUs sent to other ISs from 0 (off) to 1 (on). Thus, when an IS is overloaded, other ISs will not use the overloaded IS to forward traffic.

An IS can be in the overload state for Level 1 or Level 2 as follows:

* If an IS is in the overload state for Level 1, other Level 1 ISs stop using the overloaded IS to forward Level 1 traffic. However, the IS can still forward Level 2 traffic, if applicable.
* If an IS is in the overload state for Level 2, other Level 2 ISs stop using the overloaded IS to forward Level 2 traffic. However, the IS can still forward Level 1 traffic, if applicable.
* If an IS is in the overload state for both levels, the IS cannot forward traffic at either level.

By default, the device automatically sets the overload bit to 1 (on) in its LSPDUs to other ISs if an overload condition occurs.

The overload bit can also be set to administratively shut down IS-IS without disabling the protocol. Setting the overload bit on is useful when configuration changes are needed but you do not want to remove the device from the network. A device can also be configured to set the overload bit on for a specific number of seconds during startup, to allow IS-IS to become fully active before the device begins IS-IS routing.

Setting the overload bit

The overload bit can be configured so that when an IS is overloaded, other ISs will not use the overloaded IS to forward traffic. The following task specifies that the overload bit is set upon system startup and remains set until BGP has converged and specifies that the device that 240 seconds is the maximum time that IS-IS will wait for BGP convergence to complete.

About This Task

Procedure

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

   ```
   device# configure terminal
   ```

2. Enter the **router isis** command to enter IS-IS router configuration mode and enable IS-IS on the device.

   ```
   device(config)# router isis
   ```

3. Enter the **set-overload-bit** command with the **on-startup** and **wait-for-bgp** parameters, specifying a value, to configure the device to set the overload bit upon system startup. The overload bit remains set until BGP has converged. 86,400 seconds is the maximum time that IS-IS waits for BGP convergence to complete.

   ```
   device(config-isis-router)# set-overload-bit on-startup wait-for-bgp 86400
   ```

Example

The following example configures the overload bit on system startup until BGP has converged. The maximum time that the device waits for BGP convergence to complete is 86400 seconds.

```
device# configure terminal
device(config)# router isis
device(config-isis-router)# set-overload-bit on-startup wait-for-bgp 86400
```

Authentication

By default, a device does not authenticate packets sent to or received from an end system (ES) or other intermediate system (IS).

An authentication password can be configured using the Hashed Message Authentication codes - Message Digest 5 (HMAC-MD5) algorithm, in conformance with RFC 3567 - Intermediate System to Intermediate System (IS-IS) Cryptographic Authentication.

IS-IS authentication checking is enabled by default. When transitioning from one authentication mode to another, changing the authentication mode can cause packets to drop because only some of the routers have been reconfigured. During such a transition, it can be useful to disable IS-IS authentication checking temporarily until all devices are reconfigured and the network is stable.

Authentication can be configured globally or for a specified interface.

To configure IS-IS authentication globally you must perform the following tasks:

- Configure IS-IS Authentication Mode
- Configure IS-IS Authentication Key
- Disable IS-IS Authentication Check (optional)

To configure IS-IS authentication on a specified interface, you must perform the following tasks:

- Configure IS-IS Interface Authentication Mode for the specified interface
- Configure IS-IS Authentication Key for the specified interface
- Disable IS-IS Authentication Check for the specified interface (optional)

Configuring authentication

A device can be configured to authenticate packets sent to or received from an end system (ES) or other intermediate system (IS). The following task configures IS-IS authentication mode, an IS-IS authentication key and temporarily disables IS-IS authentication checking for Level 1 packets.

About This Task

Procedure

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

```
device# configure terminal
```

2. Enter the **router isis** command to enter IS-IS router configuration mode and enable IS-IS on the device.

```
device(config)# router isis
```

3. Enter the **auth-mode** command with the **md5** and **level-1** parameters to configure MD5 authentication for Level 1 packets.

```
device(config-isis-router)# auth-mode MD5 level-1
```

4. Enter the **auth-check disable** command with the **level-1** parameter to temporarily disable authentication checking globally for Level 1 packets.

```
device(config-isis-router)# auth-check level-1 disable
```

5. Enter the **auth-key** command with the **level-1** parameters, specifying an authentication password to configure an authentication key globally for Level 1 packets.

```
device(config-isis-router)# auth-key level-1 mysecurekey
```

Example

The following example configures IS-IS authentication mode, an IS-IS authentication key and temporarily disables IS-IS authentication checking for Level 1 packets.

```
device# configure terminal
device(config)# router isis
device(config-isis-router)# auth-mode MD5 level-1
device(config-isis-router)# auth-check level-1 disable
device(config-isis-router)# auth-key level-1 mysecurekey
```

Changing the IS-IS level globally

By default, a device can operate as both a Level 1 and Level 2 router. This task globally changes the level supported from Level 1 and Level 2 to Level 1 only.

About This Task

Procedure

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

```
device# configure terminal
```

2. Enter the **router isis** command to enter IS-IS router configuration mode and enable IS-IS on the device.

```
device(config)# router isis
```

3. Enter the **is-type** command with the **level-1** parameter to globally change the IS-IS level supported from Level-1 and Level-2 to Level-1 only.

```
device(config-isis-router)# is-type level-1
```

Example

The following example globally changes the IS-IS level supported from Level-1 and Level-2 to Level-1 only.

```
device# configure terminal
device(config)# router isis
device(config-isis-router)# is-type level-1
```

Logging adjacency changes

You can configure a device to log changes in the status of an adjacency with another IS. The following task enables the logging of adjacency changes.

About This Task

Procedure

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

   ```
   device# configure terminal
   ```

2. Enter the **router isis** command to enter IS-IS router configuration mode and enable IS-IS on the device.

   ```
   device(config)# router isis
   ```

3. Enter the **log adjacency** command to enable the logging of adjacency changes.

   ```
   device(config-isis-router)# log adjacency
   ```

Example

The following example enables the logging of adjacency changes.

```
device# configure terminal
device(config)# router isis
device(config-isis-router)# log adjacency
```

Complete Sequence Numbers PDU interval

A Complete Sequence Numbers PDU (CSNP) is a complete list of the LSPs in the Designated IS' link state database. The CSNP contains a list of all the LSPs in the database, as well as other information that helps IS neighbors determine whether their LSP databases are in sync with one another. The Designated IS sends CSNPs to the broadcast interface. Level 1 and Level 2 each have their own Designated IS.

A Partial Sequence Numbers PDU (PSNP) is a partial list of LSPs. ISs other than the Designated IS (that is, the non-Designated ISs) send PSNPs to the broadcast interface. The CSNP interval specifies how often the Designated IS sends a CSNP to the broadcast interface. Likewise, the PSNP interval specifies how often other ISs (non-Designated ISs) send a PSNP to the broadcast interface.

Configuring the CSNP interval

By default the Complete Sequence Numbers PDU (CSNP) interval is 10 seconds. The following task configures CSNP interval of 25 seconds for Level 1 and Level 2 packets.

About This Task

Procedure

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

```
device# configure terminal
```

2. Enter the **router isis** command to enter IS-IS router configuration mode and enable IS-IS on the device.

```
device(config)# router isis
```

3. Enter the **csnp-interval** command and specify a value to globally configure the CSNP interval.

```
device(config-isis-router)# csnp-interval 25
```

Example

The following example configures a CSNP interval of 25 seconds.

```
device# configure terminal
device(config)# router isis
device(config-isis-router)# csnp-interval 25
```

Changing the maximum LSP lifetime

This task changes the maximum number of seconds an unrefreshed LSP remains in a device's LSP database from the default of 1200 seconds (20 minutes) to 2400 seconds (40 minutes).

About This Task

Procedure

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

```
device# configure terminal
```

2. Enter the **router isis** command to enter IS-IS router configuration mode and enable IS-IS on the device.

```
device(config)# router isis
```

3. Enter the **max-lsp-lifetime** command and specify a value to globally configure the maximum LSP lifetime.

```
device(config-isis-router)# max-lsp-lifetime 2400
```

Example

The following example changes the maximum LSP lifetime to 2400 seconds.

```
device# configure terminal
device(config)# router isis
device(config-isis-router)# max-lsp-lifetime 2400
```

Changing the LSP refresh interval

This task changes the maximum number of seconds a device waits between sending updated LSPs to its IS-IS neighbors from the default of 900 seconds to 1800 seconds.

About This Task

Procedure

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

   ```
   device# configure terminal
   ```

2. Enter the **router isis** command to enter IS-IS router configuration mode and enable IS-IS on the device.

   ```
   device(config)# router isis
   ```

3. Enter the **lsp-refresh-interval** command and specify a value to change the LSP refresh interval.

   ```
   device(config-isis-router)# lsp-refresh-interval 1800
   ```

Example

The following example changes the LSP refresh interval to 1800 seconds.

```
device# configure terminal
device(config)# router isis
device(config-isis-router)# lsp-refresh-interval 1800
```

Changing the LSP generation interval

This task changes the minimum number of seconds the device waits between sending updated LSPs to its IS-IS neighbors from the default of 10 seconds to 60 seconds (one minute).

About This Task

Procedure

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

   ```
   device# configure terminal
   ```

2. Enter the **router isis** command to enter IS-IS router configuration mode and enable IS-IS on the device.

    ```
    device(config)# router isis
    ```

3. Enter the **lsp-gen-interval** command and specify a value to change the LSP generation interval.

    ```
    device(config-isis-router)# lsp-gen-interval 60
    ```

Example

The following example changes the LSP generation interval to 60 seconds.

```
device# configure terminal
device(config)# router isis
device(config-isis-router)# lsp-gen-interval 60
```

Changing the LSP interval and retransmit interval

The LSP interval is the rate of transmission, in milliseconds, of the LSPs. The retransmit interval is the time, in seconds, the device waits before it retransmits LSPs. This task changes the LSP interval to 45 milliseconds and changes the retransmission interval to 10 milliseconds.

About This Task

Procedure

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

    ```
    device# configure terminal
    ```

2. Enter the **router isis** command to enter IS-IS router configuration mode and enable IS-IS on the device.

    ```
    device(config)# router isis
    ```

3. Enter the **lsp-interval 45** command and specify a value to change the LSP interval from the default of 33 milliseconds.

    ```
    device(config-isis-router)# lsp-interval 42
    ```

4. Enter the **retransmit-interval** command and specify a value to change the retransmission interval from the default of 5 seconds.

    ```
    device(config-isis-router)# retransmit-interval 10
    ```

Example

The following example changes the LSP interval to 45 milliseconds and changes the retransmission interval to 10 milliseconds.

```
device# configure terminal
device(config)# router isis
```

```
device(config-isis-router)# lsp-interval 42
device(config-isis-router)# retransmit-interval 10
```

Disabling IS-IS name mapping capability

The implementation of IS-IS supports RFC 2763, which describes a mechanism for mapping IS-IS system IDs to the hostnames of the devices with those IDs. The following task disables IS-IS name mapping.

About This Task

Procedure

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

   ```
   device# configure terminal
   ```

2. Enter the **router isis** command to enter IS-IS router configuration mode and enable IS-IS on the device.

   ```
   device(config)# router isis
   ```

3. Enter the **hostname disable** command to disable IS-IS name mapping.

   ```
   device(config-isis-router)# hostname disable
   ```

Example

The following example globally disables IS-IS name mapping.

```
device# configure terminal
device(config)# router isis
device(config-isis-router)# hostname disable
```

Logging invalid LSP packets received

You can configure a device to provide logging of invalid LSP packets. The following task enables the logging of invalid LSP packets.

About This Task

Procedure

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

   ```
   device# configure terminal
   ```

2. Enter the **router isis** command to enter IS-IS router configuration mode and enable IS-IS on the device.

   ```
   device(config)# router isis
   ```

3. Enter the **log invalid-lsp-packets** command to enable the logging of invalid LSP packets.

   ```
   device(config-isis-router)# log invalid-lsp-packets
   ```

Example

The following example enables the logging of invalid LSP packets.

```
device# configure terminal
device(config)# router isis
device(config-isis-router)# log invalid-lsp-packets
```

Changing the SPF timer

Every IS maintains a Shortest Path First (SPF) tree, which is a representation of the states of each of the IS's links to ESs and other ISs. If the IS is both a Level-1 and Level-2 IS, it maintains separate SPF trees for each level. To ensure that the SPF tree remains current, the IS updates the tree at regular intervals following a change in network topology or the link state database. This task changes the maximum interval in seconds between SPF recalculations, the initial SPF calculation delay, and the hold time between the first and second SPF calculation.

About This Task

Procedure

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

   ```
   device# configure terminal
   ```

2. Enter the **router isis** command to enter IS-IS router configuration mode and enable IS-IS on the device.

   ```
   device(config)# router isis
   ```

3. Enter the **spf-interval** command with the **level-1** parameter, specifying values for the $max-wait$, $initial-wait$, and $second-wait$ parameters, to set the maximum interval in seconds between SPF recalculations, the initial SPF calculation delay, and the hold time between the first and second SPF calculation.

   ```
   device(config-isis-router)# spf-interval level-1 15 10000 15000
   ```

Example

The following example specifies that the maximum interval in seconds between SPF recalculations is 15 seconds for Level 1 packets. The initial SPF calculation delay is 10000 milliseconds and the hold time between the first and second SPF calculation is 15000 millseconds.

```
device# configure terminal
device(config)# router isis
device(config-isis-router)# spf-interval level-1 15 10000 15000
```

Configuring the IS-IS flooding mechanism

You can configure IS-IS to flood Link State PDUs to other devices in the network before running SPF, thus improving database synchronization by allowing LSP changes to be propagated to neighbors before running SPF. The following task configures the IS-IS fast-flood feature.

About This Task

Procedure

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

   ```
   device# configure terminal
   ```

2. Enter the **router isis** command to enter IS-IS router configuration mode and enable IS-IS on the device.

   ```
   device(config)# router isis
   ```

3. Enter the **fast-flood** command and specify a value to implement IS-IS fast-flood.

   ```
   device(config-isis-router)# fast-flood 10
   ```

Example

The following example configures IS-IS to flood 10 LSPs before running SPF.

```
device# configure terminal
device(config)# router isis
device(config-isis-router)# fast-flood 10
```

Configuring IS-IS PSPF exponential back-off

The following task changes the partial shortest path first (PSPF) interval, changing the maximum interval between SPF recalculations, the initial SPF calculation delay, and the hold time between the first and second SPF calculation.

About This Task

Procedure

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

   ```
   device# configure terminal
   ```

2. Enter the **router isis** command to enter IS-IS router configuration mode and enable IS-IS on the device.

   ```
   device(config)# router isis
   ```

3. Enter the **partial-spf-interval** command with the **level-1** parameter, specifying values for the *max-wait* , *initial-wait* , and *second-wait* parameters, to set the maximum interval between SPF recalculationss, the initial SPF calculation delay, and the hold time between the first and second SPF calculation.

```
device(config-isis-router)# partial-spf-interval 15 10000 15000
```

Example

The following example specifies that the maximum interval in seconds between SPF recalculations is 15 seconds. The initial SPF calculation delay is 10000 milliseconds and the hold time between the first and second SPF calculation is 15000 milliseconds.

```
device# configure terminal
device(config)# router isis
device(config-isis-router)# partial-spf-interval 15 10000 15000
```

Hello padding

By default, a device adds extra data to the end of a hello packet to make the packet the same size as the maximum length of PDU the device supports.

The padding applies to the following types of hello packets:

- ES hello (ESH PDU)
- IS hello (ISH PDU)
- IS to IS hello (IIH PDU)

The padding consists of arbitrarily valued octets. A padded hello PDU indicates the largest PDU that the device can receive. Other ISs that receive a padded hello PDU from the device can, therefore, ensure that the IS-IS PDUs they send the device are lower than the maximum value available. Similarly, if the device receives a padded hello PDU from a neighbor IS, the device knows the maximum size PDU that can be sent to the neighbor.

When padding is enabled, the maximum length of a Hello PDU sent by the device is 1514 bytes.

If you need to disable hello padding, you can do so globally or on individual interfaces. Generally, you do not need to disable padding unless a link is experiencing slow performance. If you enable or disable padding on an interface, the interface setting overrides the global setting.

Disabling hello padding globally

By default, a device adds extra data to the end of a hello packet to make the packet the same size as the maximum length of PDU the device supports. The following task disables hello padding globally.

About This Task

Procedure

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

```
device# configure terminal
```

2. Enter the **router isis** command to enter IS-IS router configuration mode and enable IS-IS on the device.

```
device(config)# router isis
```

3. Enter the **hello padding** command with the **disable** parameter to disable hello padding globally.

```
device(config-isis-router)# hello padding disable
```

Example

The following example disables hello padding globally.

```
device# configure terminal
device(config)# router isis
device(config-isis-router)# hello padding disable
```

Partial SPF optimizations

IS-IS employs certain partial SPF optimizations to make partial changes to the routing table in network change situations where the topology of the network has not changed but where there may be changes in the IP networks advertised.

These optimizations are termed partial SPF optimizations. IS-IS can be configured to perform a full SPF calculation when any network (non-topology) change occurs so that IS-IS always runs full SPF for all such network changes.

> **Note**
> If you disable the partial SPF optimizations, IS-IS automatically disables the incremental SPF optimizations and always runs full SPF. However, the reverse is not true: disabling incremental SPF optimizations does not disable partial optimizations.

Disabling partial SPF optimizations

You can configure IS-IS to perform a full SPF calculation when any network (non-topology) change occurs. This task disables partial SPF optimizations.

Procedure

1. Access global configuration mode.

```
device# configure terminal
```

2. Enter IS-IS router configuration mode and enable IS-IS on the device.

```
device(config)# router isis
```

3. Disable partial SPF optimizations.

```
device(config-isis-router)# disable-partial-spf-opt
```

Incremental SPF optimizations

In the event of certain topology changes, for instance non-local adjacency flaps, IS-IS employs incremental SPF optimizations to efficiently update the routing table. An incremental SPF is faster and takes fewer CPU cycles than a full SPF.

IS-IS can be configured to perform a full SPF calculation when any network topology change occurs so that IS-IS always runs full SPF for all such network changes.

> **Note**
> If you disable the partial SPF optimizations, IS-IS automatically disables the incremental SPF optimizations and always runs full SPF. However, the reverse is not true: disabling incremental SPF optimizations does not disable partial optimizations.

Disabling incremental shortcut SPF optimizations

You can configure IS-IS to perform a full SPF calculation when any network topology change occurs so that IS-IS always runs full SPF for all such network changes. This task disables incremental SPF optimizations so that IS-IS always runs full SPF for all such network changes.

Procedure

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

   ```
   device# configure terminal
   ```

2. Enter the **router isis** command to enter IS-IS router configuration mode and enable IS-IS on the device.

   ```
   device(config)# router isis
   ```

3. Enter the **disable-partial-spf-opt** command to disable partial SPF optimizations.

   ```
   device(config-isis-router)# disable-incremental-spf-opt
   ```

Example

The following example disables incremental SPF optimizations.

```
device# configure terminal
device(config)# router isis
device(config-isis-router)# disable-incremental-spf-opt
```

IS-IS incremental shortcut LSP SPF optimization

IS-IS can be configured to use an incremental shortcut LSP SPF optimization algorithm. Incremental shortcut LSP SPF optimization is more efficient when updating the routes in cases where the shortcut

LSP state change does not influence the topology. Incremental Shortcut LSP SPF Optimizations are enabled by default.

> **Note**
>
> If you disable the partial SPF optimizations, IS-IS automatically disables the incremental SPF optimizations and always runs full SPF. However, the reverse is not true: disabling incremental SPF optimizations does not disable partial optimizations.

> **Note**
>
> Incremental Shortcut SPF optimizations are not applicable to LSP shortcuts with metrics configured on them.

> **Note**
>
> Incremental Shortcut SPF optimizations are not applicable to LSP shortcuts with negative relative metrics configured.

> **Note**
>
> Incremental Shortcut SPF optimizations are not applicable to announced LSP shortcuts.

Disabling incremental SPF optimizations

IS-IS is configured to use an incremental shortcut LSP SPF optimization algorithm by default. This task disables incremental shortcut LSP SPF optimization.

Procedure

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

   ```
   device# configure terminal
   ```

2. Enter the **router isis** command to enter IS-IS router configuration mode and enable IS-IS on the device.

   ```
   device(config)# router isis
   ```

3. Enter the **disable-inc-stct-spf-opt** command to disable incremental shortcut LSP SPF optimization.

   ```
   device(config-isis-router)# disable-inc-stct-spf-opt
   ```

Example

The following example disables incremental shortcut LSP SPF optimization.

```
device# configure terminal
device(config)# router isis
device(config-isis-router)# disable-inc-stct-spf-opt
```

Maximum number of load sharing paths

You can change the number of paths IPv4 IS-IS can calculate and install in the IPv4 forwarding table. If you change the number of paths to one, the device does not load share multiple route paths learned from IPv4 IS-IS.

By default, IPv4 IS-IS can calculate and install eight equal-cost paths into the IPv4 forwarding table.

Changing the maximum number of load sharing paths

You can specify the number of paths IS-IS can calculate and install in the IP forwarding table. The following task specifies that the number of paths IS-IS can calculate and install in the IP forwarding table is 7.

About This Task

Procedure

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

   ```
   device# configure terminal
   ```

2. Enter the **router isis** command to enter IS-IS router configuration mode and enable IS-IS on the device.

   ```
   device(config)# router isis
   ```

3. Enter the **address-family ipv4 unicast** command to enable ISIS address-family IPv4 unicast configuration mode.

   ```
   device(config-isis-router)# address-family ipv4 unicast
   ```

4. Enter the **maximum-paths** command and specify a value to specify the number of paths IS-IS can calculate and install in the IP forwarding table.

   ```
   device(config-router-isis-ipv4u)# maximum-paths 7
   ```

Example

The following example specifies that the number of paths IS-IS can calculate and install in the IP forwarding table is 7.

```
device# configure terminal
device(config)# router isis
device(config-isis-router)# address-family ipv4 unicast
device(config-router-isis-ipv4u)# maximum-paths 7
```

Default route advertisement

By default, a device does not generate or advertise a default route to its neighboring ISs. A default route is not advertised even if the device's IPv4 route table contains a default route. You can enable the device to advertise a default route to all neighboring ISs.

> **Note**
> This feature requires the presence of a default route in the IPv4 route table.

By default, this feature originates the default route at Level 2 only. However, you can apply a route map to originate the default route to Level 1 only or at both Level 1 and Level 2. The route map must be configured before you can use the route map to configure the device to advertise the default route.

Enabling advertisement of a default route

You can enable the device to advertise a default route to all neighboring ISs. The following task enables the advertisement of a default route.

About This Task

Procedure

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

   ```
   device# configure terminal
   ```

2. Enter the **router isis** command to enter IS-IS router configuration mode and enable IS-IS on the device.

   ```
   device(config)# router isis
   ```

3. Enter the **address-family ipv4 unicast** command to enable ISIS address-family IPv4 unicast configuration mode.

   ```
   device(config-isis-router)# address-family ipv4 unicast
   ```

4. Enter the **default-information-originate** command to enable the advertisement of a default route.

   ```
   device(config-router-isis-ipv4u)# default-information-originate
   ```

Example

The following example enables the advertisement of a default route.

```
device# configure terminal
device(config)# router isis
device(config-isis-router)# address-family ipv4 unicast
device(config-router-isis-ipv4u)# default-information-originate
```

Using a route map to advertise a default route

You can use a route map to configure the device to advertise a default route to all neighboring ISs. The following task enables the advertisement of a default route if the route map "myroutemap" is satisfied.

Before You Begin

The route-map "myroutemap" must already be configured.

About This Task

Procedure

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

   ```
   device# configure terminal
   ```

2. Enter the **router isis** command to enter IS-IS router configuration mode and enable IS-IS on the device.

   ```
   device(config)# router isis
   ```

3. Enter the **address-family ipv4 unicast** command to enable ISIS address-family IPv4 unicast configuration mode.

   ```
   device(config-isis-router)# address-family ipv4 unicast
   ```

4. Enter the **default-information-originate** command with the **route-map** parameter and specify a route-map parameter to use the route map to enable the advertisement of a default route.

   ```
   device(config-router-isis-ipv4u)# default-information-originate route-map myroutemap
   ```

Example

The following example enables the advertisement of a default route if the route map "myroutemap" is satisfied.

```
device# configure terminal
device(config)# router isis
device(config-isis-router)# address-family ipv4 unicast
device(config-router-isis-ipv4u)# default-information-originate route-map myroutemap
```

IS-IS administrative distance

When the device has paths from multiple routing protocols to the same destination, it compares the administrative distances of the paths and selects the path with the lowest administrative distance to place in the IPv4 route table.

For example, if the device has a path from iBGP, from OSPF, and IPv4 IS-IS to the same destination, and all the paths are using their protocols' default administrative distances, the device selects the OSPF path, because that path has a lower administrative distance than the iBGP and IPv4 IS-IS paths.

The default IPv4 administrative distances on the device are:

- Directly connected - 0 (this value is not configurable)
- Static - 1 (applies to all static routes, including default routes)
- eBGP - 20
- OSPF - 110
- IPv4 IS-IS - 115
- iBGP - 200
- Local BGP - 200
- Unknown - 255 (the device will not use this route)

Lower administrative distances are preferred over higher distances. For example, if the device receives routes for the same network from IPv4 IS-IS and from iBGP, it will prefer the IPv4 IS-IS route by default.

Changing the administrative distance for IPv4 IS-IS

You can change the administrative distance for IPv4 IS-IS routes. The following task changes the administrative distance for the IPv4 IS-IS unicast address family.

About This Task

Procedure

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

   ```
   device# configure terminal
   ```

2. Enter the **router isis** command to enter IS-IS router configuration mode and enable IS-IS on the device.

   ```
   device(config)# router isis
   ```

3. Enter the **address-family ipv4 unicast** command to enable ISIS address-family IPv4 unicast configuration mode.

   ```
   device(config-isis-router)# address-family ipv4 unicast
   ```

4. Enter the **distance** command and specify a value to change the administrative distance for the IPv4 IS-IS unicast address family.

   ```
   device(config-router-isis-ipv4u)# distance 60
   ```

Example

The following example sets an administrative distance of 60 for the IPv4 unicast address family.

```
device# configure terminal
device(config)# router isis
device(config-isis-router)# address-family ipv4 unicast
device(config-router-isis-ipv4u)# distance 60
```

Configuring summary addresses

You can configure summary addresses to aggregate IS-IS route information. Summary addresses can enhance performance by reducing the size of the Link State database, reducing the amount of data the device needs to send to its neighbors, and reducing the CPU cycles used for IS-IS. The following task configures a summary address of 10.1.0.0 with a mask of 255.255.0.0 for Level 1 and Level 2 routes.

About This Task

Procedure

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

   ```
   device# configure terminal
   ```

2. Enter the **router isis** command to enter IS-IS router configuration mode and enable IS-IS on the device.

   ```
   device(config)# router isis
   ```

3. Enter the **address-family ipv4 unicast** command to enable ISIS address-family IPv4 unicast configuration mode.

   ```
   device(config-isis-router)# address-family ipv4 unicast
   ```

4. Enter the **summary-address** command with the **level-1** and **level-2** parameters, specifying an IPv4 address and network mask to configure a summary address and network mask for Level 1 and Level 2 routes.

   ```
   device(config-router-isis-ipv4u)# summary-address 10.1.0.0 255.255.0.0 level-1 level-2
   ```

Example

The following example configures a summary address of 10.1.0.0 with a mask of 255.255.0.0 for Level 1 and Level 2 routes.

```
device# configure terminal
device(config)# router isis
device(config-isis-router)# address-family ipv4 unicast
device(config-router-isis-ipv4u)# summary-address 10.1.0.0 255.255.0.0 level-1 level-2
```

IPv4 IS-IS route redistribution

Routes can be redistributed into IPv4 IS-IS by performing the following configuration tasks:

- Changing the default redistribution metric (optional).
- Configuring the redistribution of a particular route type into IPv4 IS-IS (mandatory).

The device can redistribute routes from the following route sources into IPv4 IS-IS:

- BGP4+
- OSPF
- Static IPv4 routes
- IPv4 routes learned from directly connected networks

The device can also can redistribute Level 1 IPv4 IS-IS routes into Level 2 IPv4 IS-IS routes, and Level 2 IPv4 IS-IS routes into Level 1 IPv4 IS-IS routes.

Route redistribution from other sources into IPv4 IS-IS is disabled by default. When you enable redistribution, the device redistributes routes only into Level 2 by default. You can specify Level 1 only, Level 2 only, or Level 1 and Level 2 when you enable redistribution.

The device automatically redistributes Level 1 routes into Level 2 routes. Thus, you do not need to enable this type of redistribution. You also can enable redistribution of Level 2 routes into Level 1 routes.

The device attempts to use the redistributed route's metric as the route's IPv4 IS-IS metric. For example, if an OSPF route has an OSPF cost of 20, the device uses 20 as the route's IPv4 IS-IS metric. The device uses the redistributed route's metric as the IPv4 IS-IS metric unless the route does not a have a valid metric. In this case, the device assigns the default metric value to the route.

An IS-IS LSP can hold around 32,000 IPv4 routes or 11,000 IPv6 routes. This number can vary depending on the prefix length of the routes.

Redistributing routes into IPv4 IS-IS

Routes can be redistributed for IPv4-IS-IS, and the routes to be redistributed can be specified.

About This Task

The redistribution of static routes into IPv4 IS-IS is configured on device1. The redistribution of connected routes into IPv4 IS-IS is configured on device2, and the connected routes to be redistributed are specified.

Procedure

1. On device1, enter the **configure terminal** command to access global configuration mode.

   ```
   device1# configure terminal
   ```

2. Enter the **router isis** command to enter IS-IS router configuration mode and enable IS-IS on the device.

   ```
   device1(config)# router isis
   ```

3. Enter the **address-family ipv4 unicast** command to enable ISIS address-family IPv4 unicast configuration mode.

   ```
   device1(config-isis-router)# address-family ipv4 unicast
   ```

4. Enter the **redistribute** command with the **static** parameter to redistribute static routes.

   ```
   device1(config-router-isis-ipv4u)# redistribute static
   ```

5. On device2, enter the **configure terminal** command to access global configuration mode.

   ```
   device2# configure terminal
   ```

6. Enter the **router isis** command to enter IS-IS router configuration mode and enable IS-IS on the device.

```
device2(config)# router isis
```

7. Enter the **address-family ipv4 unicast** command to enable ISIS address-family IPv4 unicast configuration mode.

```
device2(config-isis-router)# address-family ipv4 unicast
```

8. Enter the **redistribute** command with the **connected** and **route-map** parameters to redistribute connected routes and specify a route map.

```
device2(config-router-isis-ipv4u)# redistribute connected route-map rmap1
```

Example

The following example redistributes static routes into IPv4 IS-IS on a device.

```
device# configure terminal
device(config)# router isis
device(config-isis-router)# address-family ipv4 unicast
device(config-router-isis-ipv4u)# redistribute static
```

Example

The following example redistributes connected routes into IPv4 IS-IS on a device and specifies a route map.

```
device# configure terminal
device(config)# router isis
device(config-isis-router)# address-family ipv4 unicast
device(config-router-isis-ipv4u)# redistribute connected route-map rmap1
```

Default redistribution metric

When IPv4 IS-IS redistributes a route from another route source (such as OSPF, BGP4+, or a static IPv4 route) into IPv4 IS-IS, the route's metric value is used as its metric when the metric is not modified by a route map or metric parameter and the default redistribution metric is set to its default value of 0.

The default metric value can be changed.

> **Note**
> The implementation of IS-IS does not support the optional metric types Delay, Expense, or Error.

Changing the default redistribution metric

You can change the default metric for the IS-IS routing protocol. The following task changes the default metric value for the IPv4 unicast address family.

Procedure

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

   ```
   device# configure terminal
   ```

2. Enter the **router isis** command to enter IS-IS router configuration mode and enable IS-IS on the device.

   ```
   device(config)# router isis
   ```

3. Enter the **address-family ipv4 unicast** command to enable ISIS address-family IPv4 unicast configuration mode.

   ```
   device(config-isis-router)# address-family ipv4 unicast
   ```

4. Enter the **default-metric** command and specify a value to change the default metric value for the IPv4 unicast address family.

   ```
   device(config-router-isis-ipv4u)# default-metric 30
   ```

Example

The following example sets the default value to 30 for the IPv4 unicast address family.

```
device# configure terminal
device(config)# router isis
device(config-isis-router)# address-family ipv4 unicast
device(config-router-isis-ipv4u)# default-metric 30
```

Configuring the default link metric value globally

You can configure the metric value globally on all active IS-IS interfaces for the configured address family. The following task sets the IS-IS default link metric value for Level 1 for the IPv4 address family.

About This Task

Procedure

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

   ```
   device# configure terminal
   ```

2. Enter the **router isis** command to enter IS-IS router configuration mode and enable IS-IS on the device.

   ```
   device(config)# router isis
   ```

3. Enter the **address-family ipv4 unicast** command to enable ISIS address-family IPv4 unicast configuration mode.

```
device(config-isis-router)# address-family ipv4 unicast
```

4. Enter the **default-link-metric** command with the **level-1** parameter, and specify a value, to set the IS-IS default link metric value for Level 1 for the IPv4 address family.

```
device(config-router-isis-ipv4u)# default-link-metric level-1 30
```

Example

The following example sets the IS-IS default link metric value for Level 1 for the IPv4 address family.

```
device# configure terminal
device(config)# router isis
device(config-isis-router)# address-family ipv4 unicast
device(config-router-isis-ipv4u)# default-link-metric level-1 30
```

IS-IS metric styles

There are two types of metric styles in IS-IS:

- narrow metric
- wide metric

The range of the metric value is different for both of these styles. If there is a change in the metric-style configuration, the default-link-metric changes with it so that the new value of the default-link-metric is equal to the minimum of both the configured value and the maximum value supported for the new metric-style.

If the metric style changes from narrow metric to wide metric, no change in the value of default-link-metric occurs.

If the metric style changes from wide metric to narrow metric, and if the value of default-link-metric is greater than 63, the default-link-metric takes the value 63, as it is the maximum supported in the narrow metric.

Changing the metric style

You can increase the range of metric values supported by the device by changing the metric style to wide. The following task enables the wide metric type for Level 2 packets for the IS-IS IPv4 unicast address-family.

About This Task

Procedure

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

```
device# configure terminal
```

2. Enter the **router isis** command to enter IS-IS router configuration mode and enable IS-IS on the device.

```
device(config)# router isis
```

3. Enter the **address-family ipv4 unicast** command to enable ISIS address-family IPv4 unicast configuration mode.

```
device(config-isis-router)# address-family ipv4 unicast
```

4. Enter the **metric-style wide** command to enable the wide metric type for Level 2 packets for the IS-IS IPv4 unicast address-family.

```
device(config-router-isis-ipv4u)# metric-style wide level-2
```

Example

The following example enables the wide metric type for Level 2 packets for the IS-IS IPv4 unicast address-family.

```
device# configure terminal
device(config)# router isis
device(config-isis-router)# address-family ipv4 unicast
device(config-router-isis-ipv4u)# metric-style wide level-2
```

Enabling IS-IS for an Interface

IS-IS can be enabled for a specified interface for a device. Several commands can then be accessed that allow the configuration of IS-IS on the specified interface. This task enables IS-IS routing for an interface Ethernet.

About This Task

Procedure

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

```
device# configure terminal
```

2. Enter the **interface ethernet** command and specify an interface.

```
device(config)# interface ethernet 1/2
```

3. Enter the **ip router isis** command to enable IS-IS for the interface.

```
device(conf-if-eth-1/2)# ip router isis
```

Example

The following example enables IS-IS routing for an interface Ethernet.

```
device# configure terminal
device(config)# interface ethernet 1/2
device(conf-if-eth-1/2)# ip router isis
```

Configuring authentication on an IS-IS interface

The following task configures IS-IS authentication mode, an IS-IS authentication key, and temporarily disables IS-IS authentication checking for Level 2 packets on an IS-IS Ethernet interface.

Procedure

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

   ```
   device# configure terminal
   ```

2. Enter the **interface ethernet** command and specify an interface.

   ```
   device(config)# interface ethernet 1/2
   ```

3. Enter the **isis auth-mode** command with the **md5** and **level-2** parameters to configure MD5 authentication for Level 2 packets for the IS-IS interface.

   ```
   device(conf-if-eth-1/2)# isis auth-mode MD5 level-2
   ```

4. Enter the **isis auth-key** command with the **level-2** parameter, specifying an authentication password, to configure an authentication key for Level 2 packets for the IS-IS interface.

   ```
   device(conf-if-eth-1/2)# isis auth-key level-2 mykey
   ```

5. Enter the **isis auth-check disable** command with the **level-1** parameter to temporarily disable authentication checking for Level 1 and Level 2 packets for the IS-IS interface.

   ```
   device(conf-if-eth-1/2)# isis auth-check level-1 disable
   ```

Example

The following example configures IS-IS authentication mode, an IS-IS authentication key and temporarily disables IS-IS authentication checking for a specified IS-IS interface.

```
device# configure terminal
device(config)# interface ethernet 1/2
device(conf-if-eth-1/2)# isis auth-mode MD5 level-2
device(conf-if-eth-1/2)# isis auth-key level-2 mykey
device(conf-if-eth-1/2)# isis auth-check level-1 disable
```

Disabling hello padding for an IS-IS interface

By default, a device adds extra data to the end of a hello packet to make the packet the same size as the maximum length of PDU the device supports. The following task disables hello padding for an IS-IS Ethernet interface.

About This Task

Procedure

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

   ```
   device# configure terminal
   ```

2. Enter the **interface ethernet** command and specify an interface.

```
device(config)# interface ethernet 1/2
```

3. Enter the **isis hello padding** command with the **disable** parameter to disable hello padding for the IS-IS Ethernet interface.

```
device(conf-if-eth-1/2)# isis hello padding disable
```

Example

The following example disables hello padding for a specified IS-IS Ethernet interface.

```
device# configure terminal
device(config)# interface ethernet 1/2
device(conf-if-eth-1/2)# isis hello padding disable
```

Changing the IS-IS level on an IS-IS interface

By default, a device can operate as both a Level 1 and Level 2 router. The following task changes the level supported from Level 1 and Level 2 to Level 1 for an IS-IS interface.

About This Task

If you change the IS-IS type on an individual interface, the type you specify must also be specified globally. For example, if you globally set the type to Level 2 only, you cannot set the type on an individual interface to Level 1. The setting is accepted but does not take effect.

Procedure

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

```
device# configure terminal
```

2. Enter the **interface ethernet** command and specify an interface.

```
device(config)# interface ethernet 1/2
```

3. Enter the **isis circut-type** command with the **level-1** parameter to configure a Level 1 adjacency for the interface.

```
device(config-if-e1000-1/2)# isis circut-type level-1
```

```
device(conf-if-eth-1/2)# isis circut-type level-1
```

Example

The following example configures a Level 1 adjacency on an IS-IS Ethernet interface.

```
device# configure terminal
device(config)# interface ethernet 1/2
device(conf-if-eth-1/2)# isis circut-type level-1
```

Changing the hello multiplier for an IS-IS interface

The hello multiplier is the number by which an IS-IS interface multiplies the hello interval to obtain the hold time for Level 1 and Level 2 IS-to-IS hello PDUs. The following task changes the hello multiplier for Level 2 packets for an Ethernet interface.

Procedure

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

   ```
   device# configure terminal
   ```

2. Enter the **interface ethernet** command and specify an interface.

   ```
   device(config)# interface ethernet 1/2
   ```

3. Enter the **isis hello-mulitplier** command with the **level-2** parameter, and specify a value, to change the hello multiplier for Level 2 packets for the interface.

   ```
   device(conf-if-eth-1/2)# isis hello-mulitplier level-2 20
   ```

Example

The following example changes the hello multiplier to 20 for Level 2 packets for an Ethernet interface.

```
device# configure terminal
device(config)# interface ethernet 1/2
device(conf-if-eth-1/2)# isis hello-mulitplier level-2 20
```

Changing the hello interval for an IS-IS interface

The hello interval specifies how often an IS-IS interface sends hello messages to its IS-IS neighbors. The following task changes the hello interval for Level 1 packets to 20 for an IS-IS interface.

Procedure

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

   ```
   device# configure terminal
   ```

2. Enter the **interface ethernet** command and specify an interface.

   ```
   device(config)# interface ethernet 1/2
   ```

3. Enter the **isis hello-interval** command with the **level-1** parameter, and specify a value, to change the hello interval for Level 1 packets for the interface.

   ```
   device(conf-if-eth-1/2)# isis hello-interval level-1 20
   ```

Example

The following example changes the hello interval for Level 1 packets to 20 on a Loopback interface.

```
device# configure terminal
```

```
device(config)# interface ethernet 1/2
device(conf-if-eth-1/2)# isis hello-interval level-1 20
```

DIS Hello Interval

The DIS hello interval value is derived from the hello interval configured under the interface.

The default IS-IS hello interval is 10 sec. The default DIS hello interval is 10/3 = sec. The default value of the DIS hello interval is not changed. However, if you configure a hello interval of 20 for an interface, then the DIS hello interval for the interface becomes 20/3 =6 sec.

The DIS hello multiplier is the same as the hello multiplier configured under the interface.

IS-IS point-to-point over Ethernet

IS-IS uses its neighbor's MAC address to form an adjacency and stores the neighbors MAC address to recognize the adjacency in the future. This causes no problems with directly adjacent devices but problems can occur when adjacency is required between devices that are more than one hop away. IS-IS Point-to-Point over Ethernet accommodates an IS-IS network with this type of configuration.

When IS-IS Point-to-Point over Ethernet is used, devices that are several hops away or available through an IP GRE tunnel can form an IS-IS adjacency. IS-IS Point-to-Point over Ethernet can also be used when only two IS's are part of the broadcast network. IS-IS Point-to-Point over Ethernet is configured at the interface level of the devices that are forming an adjacency.The figure below shows two devices several hops away from each other that are configured for IS-IS adjacency.

Figure 44: IS-IS Point-to-Point configuration

Enabling IS-IS point-to-point over Ethernet

IS-IS Point-to-Point over Ethernet can be configured so that devices that are several hops away can form an IS-IS adjacency. The following task configures two devices, Device A and Device B, for IS-IS point-to-point over Ethernet.

Procedure

1. On Device A, enter the **configure terminal** command to access global configuration mode.

    ```
    deviceA# configure terminal
    ```

2. Enter the **interface ethernet** command and specify an interface.

    ```
    deviceA(config)# interface ethernet 1/2
    ```

3. Enter the **ip router isis** command to enable IS-IS for the interface.

    ```
    deviceA(conf-if-eth-1/2)# ip router isis
    ```

4. Enter the **ip address** command and specify an IP address and mask to configure a primary IP address for the interface.

    ```
    deviceA(conf-if-eth-1/2)# ip address 10.10.1.1/31
    ```

5. Enter the **isis point-to-point** command to configure IS-IS point-to-point for the Ethernet interface.

    ```
    deviceA(conf-if-eth-1/2)# isis point-to-point
    ```

6. On Device B, enter the **configure terminal** command to access global configuration mode.

    ```
    deviceB# configure terminal
    ```

7. Enter the **interface ethernet** command and specify an interface.

    ```
    deviceB(config)# interface ethernet 2/1
    ```

8. Enter the **ip router isis** command to enable IS-IS for the interface.

    ```
    deviceB(conf-if-eth-2/1)# ip router isis
    ```

9. Enter the **ip address** command and specify an IP address and mask to configure a primary IP address for the interface.

    ```
    deviceB(conf-if-eth-2/1)# ip address 10.10.1.2/31
    ```

10. Enter the **isis point-to-point** command to configure IS-IS point-to-point for the Ethernet interface.

    ```
    deviceB(conf-if-eth-2/1)# isis point-to-point
    ```

Example

The following example configures two devices, Device A and Device B, for IS-IS point-to-point over Ethernet.

```
deviceA# configure terminal
```

```
deviceA(config)# interface ethernet 1/2
deviceA(conf-if-eth-1/2)# ip router isis
deviceA(conf-if-eth-1/2)# ip address 10.10.1.1/31
deviceA(conf-if-eth-1/2)# isis point-to-point


deviceB# configure terminal
deviceB(config)# interface ethernet 2/1
deviceB(conf-if-eth-2/1)# ip router isis
deviceB(conf-if-eth-2/1)# ip address 10.10.1.2/31
deviceB(conf-if-eth-2/1)# isis point-to-point
```

Displaying IS-IS statistics

Various **show isis** commands verify information about IPv4 and IPv6 IS-IS configurations.

Before You Begin

About This Task

Use one or more of the following commands to verify IS-IS information. The commands do not have to be entered in this order.

Procedure

1. Enter the **show isis** command.

```
device# show isis

IS-IS Routing Protocol Operation State: Enabled
IS-Type: Level-1-2
System Id: 768e.f805.5812
Manual area address(es): 11
Level-1-2 Database State: On
Administrative Distance 115
Maximum Paths 8
Default redistribution metric 0
Default link metric for level-1 0 (conf)/ 10 (adv)
Default link metric for level-2 0 (conf)/ 10 (adv)
Protocol Routes Redistributed into IS-IS: None
Number of Routes Redistributed into IS-IS: 0
Level-1 Auth-mode: None
Level-2 Auth-mode: None
Metric Style Supported for Level-1: Narrow
Metric Style Supported for Level-2: Narrow
Graceful-Restart Helper Support: Enabled
ISIS Partial SPF Optimizations: Enabled
Timers:
L1 SPF: Max-wait 5s Init-wait 5000ms Second-wait 5000ms
L2 SPF: Max-wait 5s Init-wait 5000ms Second-wait 5000ms
L1 SPF is not scheduled
L2 SPF is not scheduled
PSPF: Max-wait 5000ms Init-wait 2000ms Second-wait 5000ms
PSPF is not scheduled
LSP: max-lifetime 1200s refresh-interval 900s gen-interval 10s
retransmit-interval 5s, lsp-interval 33ms
SNP: csnp-interval 10s psnp-interval 2s
Global Hello Padding: Enabled
Global Hello Padding For Point to Point Circuits: Enabled
Ptpt Three Way HandShake Mechanism: Enabled
BGP Ipv4 Converged: False BGP Ipv6 Converged: False
IS-IS Traffic Engineering Support: Disabled
No ISIS Shortcuts Configured
```

```
BFD: Disabled, BFD HoldoverInterval: 0
NSR: Disabled
LSP-SYNC: Not Globally Enabled
```

This example output displays general IS-IS information..

2. Enter the **show isis config** command.

```
device# show isis config

router isis
net 11.768e.f805.5812.00
address-family ipv4 unicast
!
address-family ipv6 unicast
!
```

This example shows the global IS-IS configuration commands that are in effect on the device..

3. Enter the **show isis counts** command.

```
device# show isis counts

Area Mismatch: 0
Max Area Mismatch: 0
System ID Length Mismatch: 0
LSP Sequence Number Skipped: 0
LSP Max Sequence Number Exceeded: 0
Level-1 Database Overload: 1
Level-2 Database Overload: 0
Our LSP Purged: 2
```

This example shows IS-IS error statistics..

4. Enter the **show isis hostname** command.

```
device# show isis hostname

Total number of entries in IS-IS Hostname Table: 1
System ID Hostname * = local IS
-----------------------------------------------------
* 768e.f805.5812 R1
```

This example shows the router-name-to-system-ID mapping table entries for the device.

5. Enter the **show isis interface** command.

```
device# show isis interface

Total number of IS-IS Interfaces: 11
Interface: Ve 301
Circuit State: UP Circuit Mode: Level 1-2
Circuit Type: BCAST Passive State: FALSE
Circuit Number: 2, MTU: 1500
Level-1 Auth-mode: NONE
Level-2 Auth-mode: NONE
Level-1 Metric: 10, Level-1 Priority: 64
Level-1 Hello Interval: 10, Level-1 Hello Multiplier: 3
Level-1 Designated IS: R1-02 Level-1 DIS Changes: 2
Level-2 Metric: 10, Level-2 Priority: 64
Level-2 Hello Interval: 10, Level-2 Hello Multiplier: 3
Level-2 Designated IS: R1-02 Level-2 DIS Changes: 2
Next IS-IS LAN Level-2 Hello in 11 seconds
Number of active Level-2 adjacencies: 0
Next IS-IS LAN Level-1 Hello in 1 seconds
Number of active Level-1 adjacencies: 0
Circuit State Changes: 1 Circuit Adjacencies State Changes: 0
```

```
Rejected Adjacencies: 0
Circuit Authentication L1 failures: 0
Circuit Authentication L2 failures: 0
Bad LSPs: 0
Control Messages Sent: 7577 Control Messages Received: 0
Hello Padding: Enabled
IP Enabled: TRUE
IP Addresses:
11.2.1.1/30
IPv6 Enabled: FALSE
MPLS TE Enabled: FALSE
BFD Enabled: FALSE
LDP-SYNC: Disabled, State:
…
```

This example shows information about IS-IS interfaces for a device.

6. Enter the **show isis interface** command, using the **brief** keyword.

```
device# show isis interface brief

Total number of IS-IS Interfaces: 11
Interface Type State Mode Passive MTU UpAdj DIS StateChg AdjStateChg
Ve 301 BCAST UP L12 FALSE 1500 0 None 1 0
Ve 302 BCAST UP L12 FALSE 1500 0 None 1 0
Ve 303 BCAST UP L12 FALSE 1500 0 None 1 0
Ve 304 BCAST UP L12 FALSE 1500 0 None 1 0
Ve 305 BCAST UP L12 FALSE 1500 0 None 1 0
Ve 306 BCAST UP L12 FALSE 1500 0 None 1 0
Ve 307 BCAST UP L12 FALSE 1500 0 None 1 0
Ve 308 BCAST UP L12 FALSE 1500 0 None 1 0
Ve 309 BCAST UP L12 FALSE 1500 0 None 1 0
Ve 310 BCAST UP L12 FALSE 1500 0 None 1 0
Lo 1 BCAST UP L12 TRUE 0 0 None 1 0
```

This example shows summarized information about IS-IS interfaces for a device.

7. Enter the **show isis neighbor** command.

```
device# show isis neighbor
```

This example shows IS-IS neighbor information.

8. Enter the **show isis spf-log** command.

```
device# show isis spf-log

ISIS Level-1 SPF Log
When Duration Nodes Count Last-Trigger-LSP Trigger
10h34m13s 0ms 1 10 R1.00-00 Interface State Change
10h34m38s 0ms 1 2 R1.00-00 Interface Config Change
10h34m43s 0ms 1 18 R1.00-00 Interface Config Change
10h34m48s 0ms 1 5 R1.00-00 Interface State Change
ISIS Level-2 SPF Log
When Duration Nodes Count Last-Trigger-LSP Trigger
10h34m13s 0ms 1 10 R1.00-00 Interface State Change
10h34m38s 0ms 1 2 R1.00-00 Interface Config Change
10h34m43s 0ms 1 18 R1.00-00 Interface Config Change
10h34m48s 0ms 1 5 R1.00-00 Interface State Change
```

This example shows IS-IS link-state packet (LSP) logging information.

9. Enter the **show isis traffic** command.

```
show isis traffic

                                Message Received      Message Sent
```

```
Level-1 Hellos              0                   44912
Level-2 Hellos              0                   44927
PTP Hellos                  0                   0
Level-1 LSP                 0                   0
Level-2 LSP                 0                   0
Level-1 CSNP                0                   0
Level-2 CSNP                0                   0
Level-1 PSNP                0                   0
Level-2 PSNP                0                   0
```

This example shows information about IS-IS packet counts..

IS-IS (IPv6)

IPv6 IS-IS single-topology mode

IPv6 IS-IS supports single-topology mode, which means that you can run IPv6 IS-IS concurrently with other network protocols such as IPv4 IS-IS throughout a topology. However, when implementing a single topology, all routers in an area (Level 1 routing) or domain (Level 2 routing) must be configured with the same set of network protocols on all its interfaces. You can configure IPv4 IS-IS only, IPv6 IS-IS only, or both IPv4 IS-IS and IPv6 IS-IS. For example, to successfully implement both IPv4 and IPv6 IS-IS in an area, you must configure both IPv4 and IPv6 IS-IS on all router interfaces in the area.
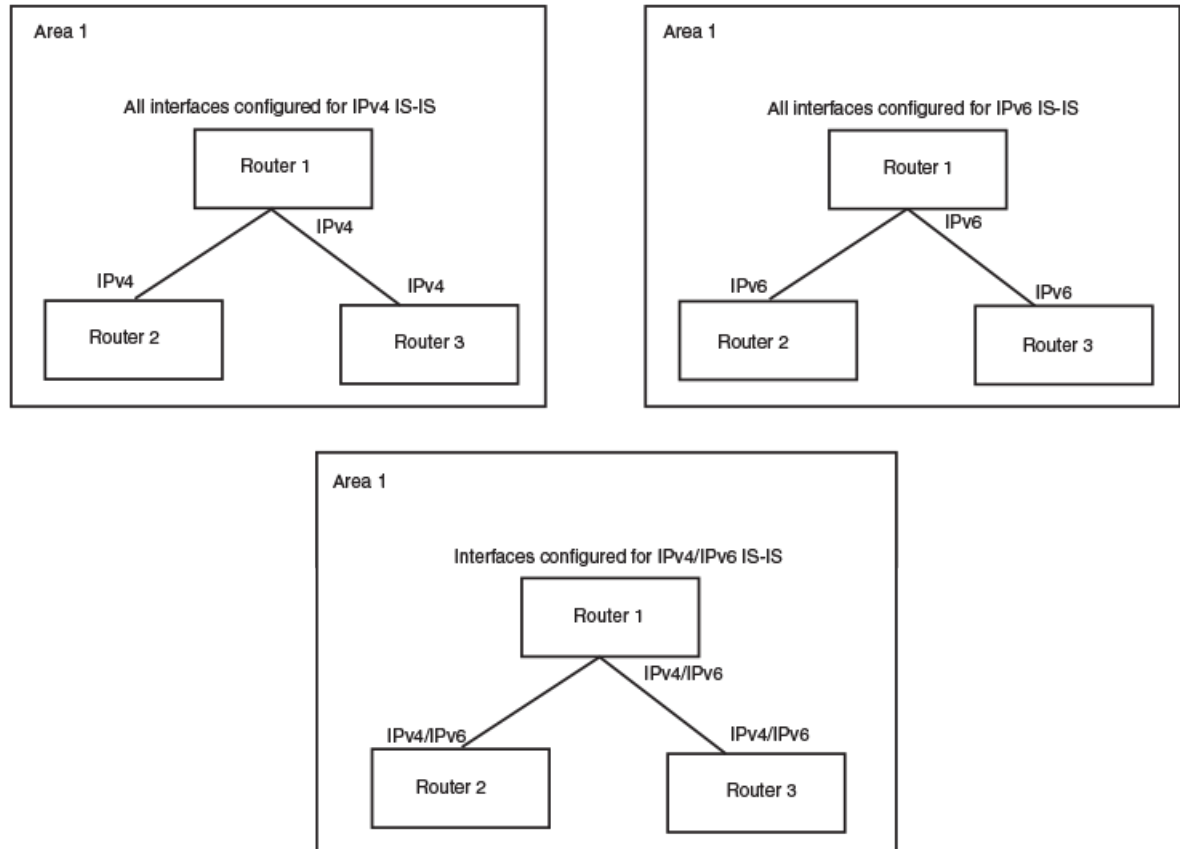
Figure 45: IPv6 IS-IS in single-topology mode

A single shortest path first (SPF) per level computes the IPv4 and IPv6 routes. The use of a single SPF indicates that both IPv4 and IPv6 IS-IS routing protocols must share a common network topology

The implementation of IPv4 IS-IS supports type, length, and value (TLV) parameters to advertise reachability to IPv4 networks. The TLVs specify the types of data, the length of the data, and the valid values for the data. IPv6 IS-IS advertises its information using new TLV parameters. The new TLV parameters for IPv6 support an extended default metric value.

In a single topology, if both IPv4 and IPv6 are configured on an interface, metric-style must be set to wide in both address families. Narrow is the default for IPv4. Wide is the default for IPv6.

IS-IS CLI levels

The CLI includes various levels of commands for IS-IS. The figure below shows these levels, including the levels used for IPv6 IS-IS.
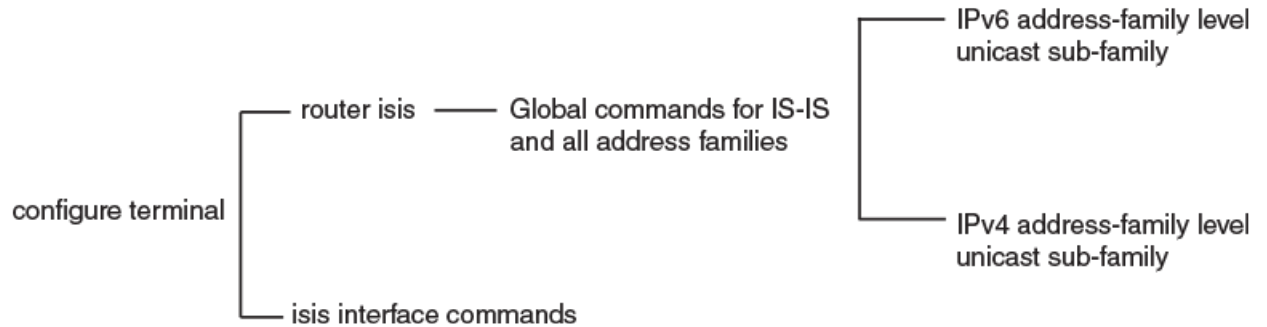
Figure 46: IPv6 IS-IS CLI levels

The IPv6 IS-IS CLI levels are as follows:

- A global level for the configuration of the IS-IS protocol. At this level, all IS-IS configurations at this level apply to IPv4 and IPv6. Enter this layer using the **router isis** command.

  ◦ Under the global level, you specify an address family. Address families separate the IS-IS configuration IPv6 and IPv4. You enter this level by entering the **address-family** command at the router IS-IS level.

  ◦ Under the address family level, you select a sub-address family, which is the type of routes for the configuration. For IS-IS, you specify **unicast**.

  ◦ An interface level.

The implementation of IPv6 IS-IS includes a new configuration level: ISIS address-family IPv6 unicast configuration mode. You enter IS-IS definitions for IPv6 IS-IS under this level. Address-family allows you to create configurations for IPv6 IS-IS unicast routes that are separate and distinct from configurations for IPv4 IS-IS unicast routes.

> **Note**
>
> Each address family configuration level allows you to access commands that apply to that particular address family only. To enable a feature in a particular address family, you must specify any associated commands for that feature in that particular address family. A feature configured in IS-IS IPv4 unicast address family configuration mode does not work for the IPv6 IS-IS unicast address family unless it is explicitly configured in ISIS address-family IPv6 unicast configuration mode. .

Enabling IPv6 IS-IS globally

When IS-IS is enabled on a device, the device enters IS-IS router configuration mode. Several commands can then be accessed that allow the configuration of IS-IS.

Procedure

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

   ```
   device# configure terminal
   ```

2. Enter the **router isis** command to enter IS-IS router configuration mode and enable IS-IS on the device.

   ```
   device(config)# router isis
   ISIS: Please configure NET!
   ```

3. Enter the **net** command and specify a NSAP address to configure a NET for IS-IS.

```
device(config-isis-router)# net 49.2211.0000.00bb.cccc.00
```

Example

The following example enables IPv6 IS-IS on a device.

```
device# configure terminal
device(config)# router isis
ISIS: Please configure NET!
device(config-isis-router)# net 49.2211.0000.00bb.cccc.00
```

Configuring the IS-IS IPv6 unicast address family

The IS-IS IPv6 unicast address family allows you to configure IPv6 IS-IS unicast settings that are separate and distinct from IPv4 IS-IS unicast settings. The following task enables ISIS IPv6 address family unicast configuration mode where a variety of IS-IS features can be configured.

About This Task

Procedure

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

```
device# configure terminal
```

2. Enter the **router isis** command to enter IS-IS router configuration mode and enable IS-IS on the device.

```
device(config)# router isis
```

3. Enter the **address-family ipv6 unicast** command to enable ISIS address-family IPv6 unicast configuration mode.

```
device(config-isis-router)# address-family ipv6 unicast
```

Example

The following example enables ISIS address-family IPv6 unicast configuration mode.

```
device# configure terminal
device(config)# router isis
device(config-isis-router)# address-family ipv6 unicast
device(config-router-isis-ipv6u)#
```

Configuring IPv6 IS-IS single topology

About This Task

If your IS-IS single topology supports both IPv6 and IPv4, you can configure both IPv6 and IPv4 on an IS-IS interface for Level 1, Level 2, or both Level 1 and Level 2. However, if you configure both IPv6 and IPv4 on an IS-IS interface, they must be configured to run on the same level. For example, you can configure IPv6 to run on Level 1 on an interface and IPv4 to also run on Level 1 on the same interface.

However, you cannot configure IPv6 to run on Level 1 on an interface and IPv4 to run to Level 2 on the same interface.

To configure an IPv6 IS-IS single topology, you must perform the tasks listed below.

Procedure

1. Globally enable IS-IS and configure at least one Network Entity Title (NET). The NET is the device's network interface with IS-IS. You can configure up to three NETs on a device.
2. Configure the desired device interfaces with an IPv6 address and enable IPv6 IS-IS on the device interfaces.
3. Configure IS-IS parameters.

Setting the overload bit

The overload bit can be configured so that when an IS is overloaded, other ISs will not use the overloaded IS to forward traffic. The following task specifies that the overload bit is set upon system startup and remains set until BGP has converged and specifies that the device that 240 seconds is the maximum time that IS-IS will wait for BGP convergence to complete.

About This Task

Procedure

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

   ```
   device# configure terminal
   ```

2. Enter the **router isis** command to enter IS-IS router configuration mode and enable IS-IS on the device.

   ```
   device(config)# router isis
   ```

3. Enter the **set-overload-bit** command with the **on-startup** and **wait-for-bgp** parameters, specifying a value, to configure the device to set the overload bit upon system startup. The overload bit remains set until BGP has converged. 86,400 seconds is the maximum time that IS-IS waits for BGP convergence to complete.

   ```
   device(config-isis-router)# set-overload-bit on-startup wait-for-bgp 86400
   ```

Example

The following example configures the overload bit on system startup until BGP has converged. The maximum time that the device waits for BGP convergence to complete is 86400 seconds.

```
device# configure terminal
device(config)# router isis
device(config-isis-router)# set-overload-bit on-startup wait-for-bgp 86400
```

Configuring authentication

A device can be configured to authenticate packets sent to or received from an end system (ES) or other intermediate system (IS). The following task configures IS-IS authentication mode, an IS-IS authentication key and temporarily disables IS-IS authentication checking for Level 1 packets.

About This Task

Procedure

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

   ```
   device# configure terminal
   ```

2. Enter the **router isis** command to enter IS-IS router configuration mode and enable IS-IS on the device.

   ```
   device(config)# router isis
   ```

3. Enter the **auth-mode** command with the **md5** and **level-1** parameters to configure MD5 authentication for Level 1 packets.

   ```
   device(config-isis-router)# auth-mode MD5 level-1
   ```

4. Enter the **auth-check disable** command with the **level-1** parameter to temporarily disable authentication checking globally for Level 1 packets.

   ```
   device(config-isis-router)# auth-check level-1 disable
   ```

5. Enter the **auth-key** command with the **level-1** parameters, specifying an authentication password to configure an authentication key globally for Level 1 packets.

   ```
   device(config-isis-router)# auth-key level-1 mysecurekey
   ```

Example

The following example configures IS-IS authentication mode, an IS-IS authentication key and temporarily disables IS-IS authentication checking for Level 1 packets.

```
device# configure terminal
device(config)# router isis
device(config-isis-router)# auth-mode MD5 level-1
device(config-isis-router)# auth-check level-1 disable
device(config-isis-router)# auth-key level-1 mysecurekey
```

Changing the IS-IS level globally

By default, a device can operate as both a Level 1 and Level 2 router. This task globally changes the level supported from Level 1 and Level 2 to Level 1 only.

About This Task

Procedure

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

```
device# configure terminal
```

2. Enter the **router isis** command to enter IS-IS router configuration mode and enable IS-IS on the device.

```
device(config)# router isis
```

3. Enter the **is-type** command with the **level-1** parameter to globally change the IS-IS level supported from Level-1 and Level-2 to Level-1 only.

```
device(config-isis-router)# is-type level-1
```

Example

The following example globally changes the IS-IS level supported from Level-1 and Level-2 to Level-1 only.

```
device# configure terminal
device(config)# router isis
device(config-isis-router)# is-type level-1
```

Logging adjacency changes

You can configure a device to log changes in the status of an adjacency with another IS. The following task enables the logging of adjacency changes.

About This Task

Procedure

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

```
device# configure terminal
```

2. Enter the **router isis** command to enter IS-IS router configuration mode and enable IS-IS on the device.

```
device(config)# router isis
```

3. Enter the **log adjacency** command to enable the logging of adjacency changes.

```
device(config-isis-router)# log adjacency
```

Example

The following example enables the logging of adjacency changes.

```
device# configure terminal
device(config)# router isis
device(config-isis-router)# log adjacency
```

Configuring the CSNP interval

By default the Complete Sequence Numbers PDU (CSNP) interval is 10 seconds. The following task configures CSNP interval of 25 seconds for Level 1 and Level 2 packets.

About This Task

Procedure

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

   ```
   device# configure terminal
   ```

2. Enter the **router isis** command to enter IS-IS router configuration mode and enable IS-IS on the device.

   ```
   device(config)# router isis
   ```

3. Enter the **csnp-interval** command and specify a value to globally configure the CSNP interval.

   ```
   device(config-isis-router)# csnp-interval 25
   ```

Example

The following example configures a CSNP interval of 25 seconds.

```
device# configure terminal
device(config)# router isis
device(config-isis-router)# csnp-interval 25
```

Changing the maximum LSP lifetime

This task changes the maximum number of seconds an unrefreshed LSP remains in a device's LSP database from the default of 1200 seconds (20 minutes) to 2400 seconds (40 minutes).

About This Task

Procedure

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

   ```
   device# configure terminal
   ```

2. Enter the **router isis** command to enter IS-IS router configuration mode and enable IS-IS on the device.

   ```
   device(config)# router isis
   ```

3. Enter the **max-lsp-lifetime** command and specify a value to globally configure the maximum LSP lifetime.

   ```
   device(config-isis-router)# max-lsp-lifetime 2400
   ```

Example

The following example changes the maximum LSP lifetime to 2400 seconds.

```
device# configure terminal
device(config)# router isis
device(config-isis-router)# max-lsp-lifetime 2400
```

Changing the LSP refresh interval

This task changes the maximum number of seconds a device waits between sending updated LSPs to its IS-IS neighbors from the default of 900 seconds to 1800 seconds.

About This Task

Procedure

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

   ```
   device# configure terminal
   ```

2. Enter the **router isis** command to enter IS-IS router configuration mode and enable IS-IS on the device.

   ```
   device(config)# router isis
   ```

3. Enter the **lsp-refresh-interval** command and specify a value to change the LSP refresh interval.

   ```
   device(config-isis-router)# lsp-refresh-interval 1800
   ```

Example

The following example changes the LSP refresh interval to 1800 seconds.

```
device# configure terminal
device(config)# router isis
device(config-isis-router)# lsp-refresh-interval 1800
```

Changing the LSP generation interval

This task changes the minimum number of seconds the device waits between sending updated LSPs to its IS-IS neighbors from the default of 10 seconds to 60 seconds (one minute).

About This Task

Procedure

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

   ```
   device# configure terminal
   ```

2. Enter the **router isis** command to enter IS-IS router configuration mode and enable IS-IS on the device.

```
device(config)# router isis
```

3. Enter the **lsp-gen-interval** command and specify a value to change the LSP generation interval.

```
device(config-isis-router)# lsp-gen-interval 60
```

Example

The following example changes the LSP generation interval to 60 seconds.

```
device# configure terminal
device(config)# router isis
device(config-isis-router)# lsp-gen-interval 60
```

Changing the LSP interval and retransmit interval

The LSP interval is the rate of transmission, in milliseconds, of the LSPs. The retransmit interval is the time, in seconds, the device waits before it retransmits LSPs. This task changes the LSP interval to 45 milliseconds and changes the retransmission interval to 10 milliseconds.

About This Task

Procedure

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

```
device# configure terminal
```

2. Enter the **router isis** command to enter IS-IS router configuration mode and enable IS-IS on the device.

```
device(config)# router isis
```

3. Enter the **lsp-interval 45** command and specify a value to change the LSP interval from the default of 33 milliseconds.

```
device(config-isis-router)# lsp-interval 42
```

4. Enter the **retransmit-interval** command and specify a value to change the retransmission interval from the default of 5 seconds.

```
device(config-isis-router)# retransmit-interval 10
```

Example

The following example changes the LSP interval to 45 milliseconds and changes the retransmission interval to 10 milliseconds.

```
device# configure terminal
device(config)# router isis
```

```
device(config-isis-router)# lsp-interval 42
device(config-isis-router)# retransmit-interval 10
```

Disabling IS-IS name mapping capability

The implementation of IS-IS supports RFC 2763, which describes a mechanism for mapping IS-IS system IDs to the hostnames of the devices with those IDs. The following task disables IS-IS name mapping.

About This Task

Procedure

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

   ```
   device# configure terminal
   ```

2. Enter the **router isis** command to enter IS-IS router configuration mode and enable IS-IS on the device.

   ```
   device(config)# router isis
   ```

3. Enter the **hostname disable** command to disable IS-IS name mapping.

   ```
   device(config-isis-router)# hostname disable
   ```

Example

The following example globally disables IS-IS name mapping.

```
device# configure terminal
device(config)# router isis
device(config-isis-router)# hostname disable
```

Logging invalid LSP packets received

You can configure a device to provide logging of invalid LSP packets. The following task enables the logging of invalid LSP packets.

About This Task

Procedure

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

   ```
   device# configure terminal
   ```

2. Enter the **router isis** command to enter IS-IS router configuration mode and enable IS-IS on the device.

   ```
   device(config)# router isis
   ```

3. Enter the **log invalid-lsp-packets** command to enable the logging of invalid LSP packets.

   ```
   device(config-isis-router)# log invalid-lsp-packets
   ```

Example

The following example enables the logging of invalid LSP packets.

```
device# configure terminal
device(config)# router isis
device(config-isis-router)# log invalid-lsp-packets
```

Changing the SPF timer

Every IS maintains a Shortest Path First (SPF) tree, which is a representation of the states of each of the IS's links to ESs and other ISs. If the IS is both a Level-1 and Level-2 IS, it maintains separate SPF trees for each level. To ensure that the SPF tree remains current, the IS updates the tree at regular intervals following a change in network topology or the link state database. This task changes the maximum interval in seconds between SPF recalculations, the initial SPF calculation delay, and the hold time between the first and second SPF calculation.

About This Task

Procedure

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

   ```
   device# configure terminal
   ```

2. Enter the **router isis** command to enter IS-IS router configuration mode and enable IS-IS on the device.

   ```
   device(config)# router isis
   ```

3. Enter the **spf-interval** command with the **level-1** parameter, specifying values for the $max-wait$, $initial-wait$, and $second-wait$ parameters, to set the maximum interval in seconds between SPF recalculations, the initial SPF calculation delay, and the hold time between the first and second SPF calculation.

   ```
   device(config-isis-router)# spf-interval level-1 15 10000 15000
   ```

Example

The following example specifies that the maximum interval in seconds between SPF recalculations is 15 seconds for Level 1 packets. The initial SPF calculation delay is 10000 milliseconds and the hold time between the first and second SPF calculation is 15000 millseconds.

```
device# configure terminal
device(config)# router isis
device(config-isis-router)# spf-interval level-1 15 10000 15000
```

Configuring the IS-IS flooding mechanism

You can configure IS-IS to flood Link State PDUs to other devices in the network before running SPF, thus improving database synchronization by allowing LSP changes to be propagated to neighbors before running SPF. The following task configures the IS-IS fast-flood feature.

About This Task

Procedure

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

   ```
   device# configure terminal
   ```

2. Enter the **router isis** command to enter IS-IS router configuration mode and enable IS-IS on the device.

   ```
   device(config)# router isis
   ```

3. Enter the **fast-flood** command and specify a value to implement IS-IS fast-flood.

   ```
   device(config-isis-router)# fast-flood 10
   ```

Example

The following example configures IS-IS to flood 10 LSPs before running SPF.

```
device# configure terminal
device(config)# router isis
device(config-isis-router)# fast-flood 10
```

Configuring IS-IS PSPF exponential back-off

The following task changes the partial shortest path first (PSPF) interval, changing the maximum interval between SPF recalculations, the initial SPF calculation delay, and the hold time between the first and second SPF calculation.

About This Task

Procedure

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

   ```
   device# configure terminal
   ```

2. Enter the **router isis** command to enter IS-IS router configuration mode and enable IS-IS on the device.

   ```
   device(config)# router isis
   ```

3. Enter the **partial-spf-interval** command with the **level-1** parameter, specifying values for the *max-wait* , *initial-wait* , and *second-wait* parameters, to set the maximum interval between SPF recalculationss, the initial SPF calculation delay, and the hold time between the first and second SPF calculation.

```
device(config-isis-router)# partial-spf-interval 15 10000 15000
```

Example

The following example specifies that the maximum interval in seconds between SPF recalculations is 15 seconds. The initial SPF calculation delay is 10000 milliseconds and the hold time between the first and second SPF calculation is 15000 milliseconds.

```
device# configure terminal
device(config)# router isis
device(config-isis-router)# partial-spf-interval 15 10000 15000
```

Disabling hello padding globally

By default, a device adds extra data to the end of a hello packet to make the packet the same size as the maximum length of PDU the device supports. The following task disables hello padding globally.

About This Task

Procedure

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

```
device# configure terminal
```

2. Enter the **router isis** command to enter IS-IS router configuration mode and enable IS-IS on the device.

```
device(config)# router isis
```

3. Enter the **hello padding** command with the **disable** parameter to disable hello padding globally.

```
device(config-isis-router)# hello padding disable
```

Example

The following example disables hello padding globally.

```
device# configure terminal
device(config)# router isis
device(config-isis-router)# hello padding disable
```

Disabling partial SPF optimizations

You can configure IS-IS to perform a full SPF calculation when any network (non-topology) change occurs. This task disables partial SPF optimizations.

Procedure

1. Access global configuration mode.

```
device# configure terminal
```

2. Enter IS-IS router configuration mode and enable IS-IS on the device.

```
device(config)# router isis
```

3. Disable partial SPF optimizations.

```
device(config-isis-router)# disable-partial-spf-opt
```

Disabling incremental SPF optimizations

IS-IS is configured to use an incremental shortcut LSP SPF optimization algorithm by default. This task disables incremental shortcut LSP SPF optimization.

Procedure

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

```
device# configure terminal
```

2. Enter the **router isis** command to enter IS-IS router configuration mode and enable IS-IS on the device.

```
device(config)# router isis
```

3. Enter the **disable-inc-stct-spf-opt** command to disable incremental shortcut LSP SPF optimization.

```
device(config-isis-router)# disable-inc-stct-spf-opt
```

Example

The following example disables incremental shortcut LSP SPF optimization.

```
device# configure terminal
device(config)# router isis
device(config-isis-router)# disable-inc-stct-spf-opt
```

Disabling incremental shortcut SPF optimizations

You can configure IS-IS to perform a full SPF calculation when any network topology change occurs so that IS-IS always runs full SPF for all such network changes. This task disables incremental SPF optimizations so that IS-IS always runs full SPF for all such network changes.

Procedure

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

   ```
   device# configure terminal
   ```

2. Enter the **router isis** command to enter IS-IS router configuration mode and enable IS-IS on the device.

   ```
   device(config)# router isis
   ```

3. Enter the **disable-partial-spf-opt** command to disable partial SPF optimizations.

   ```
   device(config-isis-router)# disable-incremental-spf-opt
   ```

Example

The following example disables incremental SPF optimizations.

```
device# configure terminal
device(config)# router isis
device(config-isis-router)# disable-incremental-spf-opt
```

Maximum number of load sharing paths

You can change the number of paths IPv6 IS-IS can calculate and install in the IPv6 forwarding table. If you change the number of paths to one, the device does not load share multiple route paths learned from IPv4 IS-IS.

By default, IPv6 IS-IS can calculate and install eight equal-cost paths into the IPv4 forwarding table.

Changing the maximum number of load sharing paths

You can specify the number of paths IS-IS can calculate and install in the IP forwarding table. The following task specifies that the number of paths IS-IS can calculate and install in the IP forwarding table is 5 for the IS-IS IPv6 unicast address family.

About This Task

Procedure

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

   ```
   device# configure terminal
   ```

2. Enter the **router isis** command to enter IS-IS router configuration mode and enable IS-IS on the device.

```
device(config)# router isis
```

3. Enter the **address-family ipv6 unicast** command to enable ISIS address-family IPv6 unicast configuration mode.

```
device(config-isis-router)# address-family ipv6 unicast
```

4. Enter the **maximum-paths** command and specify a value to specify the number of paths IS-IS can calculate and install in the IP forwarding table.

```
device(config-router-isis-ipv6u)# maximum-paths 5
```

Example

The following example specifies that the number of paths IS-IS can calculate and install in the IP forwarding table is 5 for the IS-IS IPv6 unicast address family.

```
device# configure terminal
device(config)# router isis
device(config-isis-router)# address-family ipv6 unicast
device(config-router-isis-ipv6u)# maximum-paths 5
```

Default route advertisement

By default, a device does not generate or advertise a default route to its neighboring ISs. A default route is not advertised even if the device's IPv6 route table contains a default route. You can enable the device to advertise a default route to all neighboring ISs.

> **Note**
> This feature requires the presence of a default route in the IPv6 route table.

You can also use a route map to configure the device to advertise a default route. The route map must be configured before you can use the route map to configure the device to advertise the default route.

Enabling advertisement of a default route

You can enable the device to advertise a default route to all neighboring ISs. The following task enables the advertisement of a default route.

About This Task

Procedure

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

```
device# configure terminal
```

2. Enter the **router isis** command to enter IS-IS router configuration mode and enable IS-IS on the device.

```
device(config)# router isis
```

3. Enter the **address-family ipv6 unicast** command to enable ISIS address-family IPv6 unicast configuration mode.

```
device(config-isis-router)# address-family ipv6 unicast
```

4. Enter the **default-information-originate** command to enable the advertisement of a default route.

```
device(config-router-isis-ipv6u)# default-information-originate
```

Example

The following example enables the advertisement of a default route for the IS-IS IPv6 unicast address-family.

```
device# configure terminal
device(config)# router isis
device(config-isis-router)# address-family ipv6 unicast
device(config-router-isis-ipv6u)# default-information-originate
```

Using a route map to advertise a default route

You can use a route map to configure the device to advertise a default route to all neighboring ISs. The following task enables the advertisement of a default route if the route map "myroutemap" is satisfied for the IS-IS IPv6 address family.

Before You Begin

The route-map "myroutemap" must already be configured.

About This Task

Procedure

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

```
device# configure terminal
```

2. Enter the **router isis** command to enter IS-IS router configuration mode and enable IS-IS on the device.

```
device(config)# router isis
```

3. Enter the **address-family ipv6 unicast** command to enable ISIS address-family IPv6 unicast configuration mode.

```
device(config-isis-router)# address-family ipv6 unicast
```

4. Enter the **default-information-originate** command with the **route-map** parameter and specify a route-map parameter to use the route map to enable the advertisement of a default route.

```
device(config-router-isis-ipv6u)# default-information-originate route-map myroutemap
```

Example

The following example enables the advertisement of a default route if the route map "myroutemap" is satisfied for the IPv6 address family.

```
device# configure terminal
device(config)# router isis
device(config-isis-router)# address-family ipv6 unicast
device(config-router-isis-ipv6u)# default-information-originate route-map myroutemap
```

IPv6 IS-IS administrative distance

When the device has paths from multiple routing protocols to the same destination, it compares the administrative distances of the paths and selects the path with the lowest administrative distance to place in the IPv6 route table.

For example, if the device has a path from iBGP, from OSPFv3, and IPv6 IS-IS to the same destination, and all the paths are using their protocols' default administrative distances, the device selects the OSPFv3 path, because that path has a lower administrative distance than the iBGP and IPv6 IS-IS paths.

The default IPv6 administrative distances on the device are:

- Directly connected - 0 (this value is not configurable)
- Static - 1 (applies to all static routes, including default routes)
- eBGP - 20
- OSPFv3 - 110
- IPv6 IS-IS - 115
- iBGP - 200
- Local BGP - 200
- Unknown - 255 (the device will not use this route)

Lower administrative distances are preferred over higher distances. For example, if the device receives routes for the same network from IPv6 IS-IS and from iBGP, it will prefer the IPv6 IS-IS route by default.

Changing the administrative distance for IPv6 IS-IS

You can change the administrative distance for IPv6 IS-IS routes. The following task changes the administrative distance for the IS-IS IPv6 unicast address family.

About This Task

Procedure

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

   ```
   device# configure terminal
   ```

2. Enter the **router isis** command to enter IS-IS router configuration mode and enable IS-IS on the device.

   ```
   device(config)# router isis
   ```

3. Enter the **address-family ipv6 unicast** command to enable ISIS address-family IPv6 unicast configuration mode.

   ```
   device(config-isis-router)# address-family ipv6 unicast
   ```

4. Enter the **distance** command and specify a value to change the administrative distance for the IPv6 IS-IS unicast address family.

   ```
   device(config-router-isis-ipv6u)# distance 60
   ```

Example

The following example sets an administrative distance of 60 for the IPv6 unicast address family.

```
device# configure terminal
device(config)# router isis
device(config-isis-router)# address-family ipv6 unicast
device(config-router-isis-ipv6u)# distance 60
```

Configuring summary prefixes

You can configure summary prefixes to aggregate IS-IS IPv6 route information. Summary prefixes can enhance performance by reducing the size of the Link State database, reducing the amount of data the device needs to send to its neighbors, and reducing the CPU cycles used for IPv6 IS-IS. The following task configures a summary prefix 2001:db8::/32 for Level 1 routes.

About This Task

Procedure

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

   ```
   device# configure terminal
   ```

2. Enter the **router isis** command to enter IS-IS router configuration mode and enable IS-IS on the device.

   ```
   device(config)# router isis
   ```

3. Enter the **address-family ipv6 unicast** command to enable ISIS address-family IPv6 unicast configuration mode.

   ```
   device(config-isis-router)# address-family ipv6 unicast
   ```

4. Enter the **summary-prefix** command with the **level-1** parameter, specifying an IPv6 address and network mask, to configure a summary prefix and network mask for Level 1 routes.

```
device(config-router-isis-ipv6u)# summary-prefix 2001:db8::/32 level-1
```

Example

The following example configures asummary-prefix of 2001:db8::/32 for Level 1 routes.

```
device# configure terminal
device(config)# router isis
device(config-isis-router)# address-family ipv6 unicast
device(config-router-isis-ipv6u)# summary-prefix 2001:db8::/32 level-1
```

Redistributing routes into IPv6 IS-IS

Routes can be redistributed into IPv6 IS-IS by performing the following configuration tasks:

- Changing the default redistribution metric (optional).
- Configuring the redistribution of a particular route type into IPv6 IS-IS (mandatory).

The device can redistribute routes from the following route sources into IPv6 IS-IS:

- BGP4+
- OSPFv3
- Static IPv6 routes
- IPv6 routes learned from directly connected networks

The device can also can redistribute Level 1 IPv6 IS-IS routes into Level 2 IPv6 IS-IS routes, and Level 2 IPv6 IS-IS routes into Level 1 IPv6 IS-IS routes.

Route redistribution from other sources into IPv6 IS-IS is disabled by default. When you enable redistribution, the device redistributes routes only into Level 2 by default. You can specify Level 1 only, Level 2 only, or Level 1 and Level 2 when you enable redistribution.

The device automatically redistributes Level 1 routes into Level 2 routes. Thus, you do not need to enable this type of redistribution. You also can enable redistribution of Level 2 routes into Level 1 routes.

The device attempts to use the redistributed route's metric as the route's IPv6 IS-IS metric. For example, if an OSPFv3 route has an OSPF cost of 20, the device uses 20 as the route's IPv6 IS-IS metric. The device uses the redistributed route's metric as the IPv6 IS-IS metric unless the route does not a have a valid metric. In this case, the device assigns the default metric value to the route.

An IS-IS LSP can hold around 32,000 IPv4 routes or 11,000 IPv6 routes. This number can vary depending on the prefix length of the routes.

Redistributing routes into IPv6 IS-IS

Routes can be redistributed for IPv6-IS-IS, and the routes to be redistributed can be specified.

About This Task

The redistribution of static routes into IPv6 IS-IS is configured on device1. The redistribution of BGP routes into IPv6 IS-IS is configured on device2, and the BGP routes to be redistributed are specified.

Procedure

1. On device1, enter the **configure terminal** command to access global configuration mode.

   ```
   device1# configure terminal
   ```

2. Enter the **router isis** command to enter IS-IS router configuration mode and enable IS-IS on the device.

   ```
   device1(config)# router isis
   ```

3. Enter the **address-family ipv6 unicast** command to enable ISIS address-family IPv6 unicast configuration mode.

   ```
   device1(config-isis-router)# address-family ipv6 unicast
   ```

4. Enter the **redistribute** command with the **static** parameter to redistribute static routes.

   ```
   device1(config-router-isis-ipv6u)# redistribute static
   ```

5. On device2, enter the **configure terminal** command to access global configuration mode.

   ```
   device2# configure terminal
   ```

6. Enter the **router isis** command to enter IS-IS router configuration mode and enable IS-IS on the device.

   ```
   device2(config)# router isis
   ```

7. Enter the **address-family ipv6 unicast** command to enable ISIS address-family IPv6 unicast configuration mode.

   ```
   device2(config-isis-router)# address-family ipv6 unicast
   ```

8. Enter the **redistribute** command with the **bgp** and **route-map** parameters to redistribute BGP routes and specify a route map.

   ```
   device2(config-router-isis-ipv6u)# redistribute bgp route-map myroutemap
   ```

Example

The following example redistributes static routes into IPv6 IS-IS on a device.

```
device# configure terminal
device(config)# router isis
device(config-isis-router)# address-family ipv6 unicast
device(config-router-isis-ipv6u)# redistribute static
```

Example

The following example redistributes BGP routes into IPv6 IS-IS on a device and specifies a route map.

```
device# configure terminal
device(config)# router isis
device(config-isis-router)# address-family ipv6 unicast
device(config-router-isis-ipv6u)# redistribute bgp route-map myroutemap
```

Default redistribution metric

When IPv6 IS-IS redistributes a route from another route source (such as OSPv3F, BGP4+, or a static IPv6 route) into IPv6 IS-IS, the route's metric value is used as its metric when the metric is not modified by a route map or metric parameter and the default redistribution metric is set to its default value of 0.

The default metric value can be changed.

> **Note**
> The implementation of IS-IS does not support the optional metric types Delay, Expense, or Error.

Changing the default redistribution metric

You can change the default metric for the IS-IS routing protocol. The following task changes the default metric value for the IPv6 unicast address family.

Procedure

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

   ```
   device# configure terminal
   ```

2. Enter the **router isis** command to enter IS-IS router configuration mode and enable IS-IS on the device.

   ```
   device(config)# router isis
   ```

3. Enter the **address-family ipv6 unicast** command to enable ISIS address-family IPv6 unicast configuration mode.

   ```
   device(config-isis-router)# address-family ipv6 unicast
   ```

4. Enter the **default-metric** command and specify a value to change the default metric value for the IPv6 unicast address family.

   ```
   device(config-router-isis-ipv6u)# default-metric 45
   ```

Example

The following example sets the default value to 45 for the IPv6 unicast address family.

```
device# configure terminal
device(config)# router isis
```

```
device(config-isis-router)# address-family ipv6 unicast
device(config-router-isis-ipv6u)# default-metric 45
```

Configuring the default link metric value globally

You can configure the metric value globally on all active IS-IS interfaces for the IS-IS IPv6 address family. The following task sets the IS-IS default link metric value for Level 1 for the IPv6 address family.

Procedure

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

   ```
   device# configure terminal
   ```

2. Enter the **router isis** command to enter IS-IS router configuration mode and enable IS-IS on the device.

   ```
   device(config)# router isis
   ```

3. Enter the **address-family ipv6 unicast** command to enable ISIS address-family IPv6 unicast configuration mode.

   ```
   device(config-isis-router)# address-family ipv6 unicast
   ```

4. Enter the **default-link-metric** command with the **level-1** parameter, and specify a value, to set the IS-IS default link metric value for Level 1 for the IPv6 address family.

   ```
   device(config-router-isis-ipv6u)# default-link-metric level-1 45
   ```

Example

The following example sets the IS-IS default link metric value for Level 1 for the IPv6 address family.

```
device# configure terminal
device(config)# router isis
device(config-isis-router)# address-family ipv6 unicast
device(config-router-isis-ipv6u)# default-link-metric level-1 45
```

IPv6 metric behavior with multi-topology configuration

The default-link-metric for IPv6 depends upon the multi-topology configuration.

No multi-topology: The IPv6 default-link-metric is the same as that configured for IPv4 address-family.

Multi-topology: The IPv6 default-link-metric is equal to the value configured for IPv6 address-family.

Multi-topology transition: The IPv6 default-link-metric will be equal to the value configured for IPv6 address-family.

IS-IS metric styles

There are two types of metric styles in IS-IS:

- narrow metric
- wide metric

The range of the metric value is different for both of these styles. If there is a change in the metric-style configuration, the default-link-metric changes with it so that the new value of the default-link-metric is equal to the minimum of both the configured value and the maximum value supported for the new metric-style.

If the metric style changes from narrow metric to wide metric, no change in the value of default-link-metric occurs.

If the metric style changes from wide metric to narrow metric, and if the value of default-link-metric is greater than 63, the default-link-metric takes the value 63, as it is the maximum supported in the narrow metric.

IPv6 protocol-support consistency checks

An IS-IS single topology must be configured to run the same set of network protocols (IPv4 IS-IS only, IPv6 IS-IS only, or both IPv4 IS-IS and IPv6 IS-IS).

By default, IS-IS performs consistency checks on hello packets. If a hello packet does not have the same configured network protocols, IS-IS rejects the packet. For example, a hello packet from a device running IPv4 and IPv6 IS-IS will be rejected by a device running either IPv4 IS-IS only or IPv6 IS-IS only, and the two devices will not become adjacent.

You can configure two devices running different sets of network protocols to form an adjacency.

Enabling IS-IS and assigning an IPv6 address to an interface

IS-IS can be enabled for a specified interface for a device. Several commands can then be accessed that allow the configuration of IS-IS on the specified interface. This task enables IS-IS routing for an interface Ethernet.

About This Task

Procedure

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

   ```
   device# configure terminal
   ```

2. Enter the **interface ethernet** command and specify an interface.

   ```
   device(config)# interface ethernet 1/2
   ```

3. Enter the **ipv6 address** command with the **eui-64** parameter, specifying an IPv6 address, to assign the IPv6 address to the interface and configure the IPv6 address with an EUI-64 interface ID in the low-order 64 bits.

   ```
   device(conf-if-eth-1/2)# ipv6 address 2001:db8:12d:1300::/64 eui-64
   ```

4. Enter the **ipv6 router isis** command to enable IS-IS for the interface.

```
device(conf-if-eth-1/2)# ipv6 router isis
```

Example

The following example enables IS-IS routing for an interface Ethernet and assigns an IPv6 address to the interface.

```
device# configure terminal
device(config)# interface ethernet 1/2
device(conf-if-eth-1/2)# ipv6 address 2001:db8:12d:1300::/64 eui-64
device(conf-if-eth-1/2)# ipv6 router isis
```

Configuring authentication on an IS-IS interface

The following task configures IS-IS authentication mode, an IS-IS authentication key, and temporarily disables IS-IS authentication checking for Level 2 packets on an IS-IS Ethernet interface.

Procedure

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

```
device# configure terminal
```

2. Enter the **interface ethernet** command and specify an interface.

```
device(config)# interface ethernet 1/2
```

3. Enter the **isis auth-mode** command with the **md5** and **level-2** parameters to configure MD5 authentication for Level 2 packets for the IS-IS interface.

```
device(conf-if-eth-1/2)# isis auth-mode MD5 level-2
```

4. Enter the **isis auth-key** command with the **level-2** parameter, specifying an authentication password, to configure an authentication key for Level 2 packets for the IS-IS interface.

```
device(conf-if-eth-1/2)# isis auth-key level-2 mykey
```

5. Enter the **isis auth-check disable** command with the **level-1** parameter to temporarily disable authentication checking for Level 1 and Level 2 packets for the IS-IS interface.

```
device(conf-if-eth-1/2)# isis auth-check level-1 disable
```

Example

The following example configures IS-IS authentication mode, an IS-IS authentication key and temporarily disables IS-IS authentication checking for a specified IS-IS interface.

```
device# configure terminal
device(config)# interface ethernet 1/2
device(conf-if-eth-1/2)# isis auth-mode MD5 level-2
device(conf-if-eth-1/2)# isis auth-key level-2 mykey
device(conf-if-eth-1/2)# isis auth-check level-1 disable
```

Disabling hello padding for an IS-IS interface

By default, a device adds extra data to the end of a hello packet to make the packet the same size as the maximum length of PDU the device supports. The following task disables hello padding for an IS-IS Ethernet interface.

About This Task

Procedure

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

   ```
   device# configure terminal
   ```

2. Enter the **interface ethernet** command and specify an interface.

   ```
   device(config)# interface ethernet 1/2
   ```

3. Enter the **isis hello padding** command with the **disable** parameter to disable hello padding for the IS-IS Ethernet interface.

   ```
   device(conf-if-eth-1/2)# isis hello padding disable
   ```

Example

The following example disables hello padding for a specified IS-IS Ethernet interface.

```
device# configure terminal
device(config)# interface ethernet 1/2
device(conf-if-eth-1/2)# isis hello padding disable
```

Changing the IS-IS level on an IS-IS interface

By default, a device can operate as both a Level 1 and Level 2 router. The following task changes the level supported from Level 1 and Level 2 to Level 1 for an IS-IS interface.

About This Task
If you change the IS-IS type on an individual interface, the type you specify must also be specified globally. For example, if you globally set the type to Level 2 only, you cannot set the type on an individual interface to Level 1. The setting is accepted but does not take effect.

Procedure

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

   ```
   device# configure terminal
   ```

2. Enter the **interface ethernet** command and specify an interface.

   ```
   device(config)# interface ethernet 1/2
   ```

3. Enter the **isis circut-type** command with the **level-1** parameter to configure a Level 1 adjacency for the interface.

```
device(config-if-e1000-1/2)# isis circut-type level-1

device(conf-if-eth-1/2)# isis circut-type level-1
```

Example

The following example configures a Level 1 adjacency on an IS-IS Ethernet interface.

```
device# configure terminal
device(config)# interface ethernet 1/2
device(conf-if-eth-1/2)# isis circut-type level-1
```

Changing the hello multiplier for an IS-IS interface

The hello multiplier is the number by which an IS-IS interface multiplies the hello interval to obtain the hold time for Level 1 and Level 2 IS-to-IS hello PDUs. The following task changes the hello multiplier for Level 2 packets for an Ethernet interface.

Procedure

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

```
device# configure terminal
```

2. Enter the **interface ethernet** command and specify an interface.

```
device(config)# interface ethernet 1/2
```

3. Enter the **isis hello-mulitplier** command with the **level-2** parameter, and specify a value, to change the hello multiplier for Level 2 packets for the interface.

```
device(conf-if-eth-1/2)# isis hello-mulitplier level-2 20
```

Example

The following example changes the hello multiplier to 20 for Level 2 packets for an Ethernet interface.

```
device# configure terminal
device(config)# interface ethernet 1/2
device(conf-if-eth-1/2)# isis hello-mulitplier level-2 20
```

Changing the hello interval for an IS-IS interface

The hello interval specifies how often an IS-IS interface sends hello messages to its IS-IS neighbors. The following task changes the hello interval for Level 1 packets to 20 for an IS-IS interface.

Procedure

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

```
device# configure terminal
```

2. Enter the **interface ethernet** command and specify an interface.

```
device(config)# interface ethernet 1/2
```

3. Enter the **isis hello-interval** command with the **level-1** parameter, and specify a value, to change the hello interval for Level 1 packets for the interface.

```
device(conf-if-eth-1/2)# isis hello-interval level-1 20
```

Example

The following example changes the hello interval for Level 1 packets to 20 on a Loopback interface.

```
device# configure terminal
device(config)# interface ethernet 1/2
device(conf-if-eth-1/2)# isis hello-interval level-1 20
```

Changing the metric added to advertised routes for an IS-IS interface

When a device originates an IS-IS route or calculates a route, it adds a metric (cost) to the route. Each IS-IS interface has a separate metric value. The following task changes the IS-IS metric for an IS-IS Ethernet interface.

About This Task

Procedure

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

```
device# configure terminal
```

2. Enter the **interface ethernet** command and specify an interface.

```
device(config)# interface ethernet 1/2
```

3. Enter the **isis metric** command with the **level-1** parameter and specify a value to change the metric for the interface, specifying Level 1 packets.

```
device(conf-if-eth-1/2)# isis metric 38 level-1
```

Example

The following example changes the metric for an IS-IS Ethernet interface to 38, specifying Level 1 packets. .

```
device# configure terminal
device(config)# interface ethernet 1/2
device(conf-if-eth-1/2)# isis metric 38 level-1
```

Disabling IPv6 protocol-support consistency checks

By default, IS-IS performs consistency checks on hello packets. If a hello packet does not have the same configured network protocols, IS-IS rejects the packet. The following task disables the IS-IS IPv6

protocol-support consistency checks so that two devices running different sets of network protocols can form an adjacency.

Procedure

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

```
device# configure terminal
```

2. Enter the **router isis** command to enter IS-IS router configuration mode and enable IS-IS on the device.

```
device(config)# router isis
```

3. Enter the **address-family ipv6 unicast** command to enable ISIS address-family IPv6 unicast configuration mode.

```
device(config-isis-router)# address-family ipv6 unicast
```

4. Enter the **disable-adjacency-check** command to disable IS-IS IPv6 protocol-support consistency checks.

```
device(config-router-isis-ipv6u)# disable-adjacency-check
```

Example

The following example disables IS-IS IPv6 protocol-support consistency checks..

```
device# configure terminal
device(config)# router isis
device(config-isis-router)# address-family ipv6 unicast
device(config-router-isis-ipv6u)# disable-adjacency-check
```

IPv6 IS-IS Multi-Topology

IPv6 IS-IS supports Multi-Topology (MT) mode, which allows you to configure both IPv4 and IPv6 topologies on the router interfaces in an area or a domain. However, when implementing an MT, all routers in an area (Level 1 routing) or a domain (Level 2 routing) can be configured with a set of independent topologies on all their interfaces, even on loopback interfaces. All routers in an area or a domain use the same type of IPv6 support, either single-topology or MT. In a network, the Shortest Path First (SPF) is calculated for each configured topology.

The figure below depicts a non-congruent topology with IPv6 IS-IS MT enabled. Router 1 is an IPv4 and IPv6 dual stack router, Router 2 is an IPv6 router, and Router 3 is an IPv4 router. All the routers (Router 1, Router 2, and Router 3) in the Area 1 are configured with a set of independent topologies.

Area 1

All interfaces are IPv6 IS-IS MT enabled

Router 1

IPv4 and IPv6

IPv6   IPv4

Router 2   Router 3
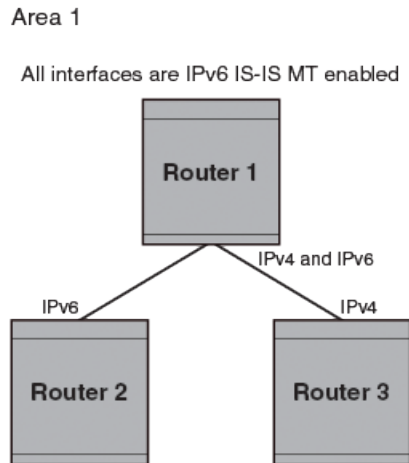
Figure 47: IS-IS non-congruent topology

Configuration considerations for IPv6 IS-IS MT

The following configuration considerations apply for IPv6 IS-IS MT

- The wide metric style must be configured before enabling IPv6 IS-IS MT.
- IPv4, IPv6, or IPv4 and IPv6 configured on the same interface must run on the same IS-IS level.
- Enabling or disabling IPv6 IS-IS MT clears all adjacencies, LSP databases, and IPv6 IS-IS routes.
- All routers on a point-to-point or a broadcast interface must support at least one common topology (IPv4 or IPv6), when MT is enabled.

Migrating to IPv6 IS-IS MT

About This Task

The following steps must be performed to migrate from a non-MT environment to an MT environment.

Procedure

1. Assume that the entire network is not an IPv6 IS-IS MT environment, and ensure that all the routes are correct.
2. Use the `multi-topology` command with the `transition` parameter to enable transition mode on each router one by one, and ensure that all the routes are correct.
3. After all the routers are in transition mode, use the `no multi-topology` command with the `transition` parameter to disable transition mode on each router one by one. Ensure that all the routes are correct.
4. Change the topology to make IPv4 and IPv6 different.

Maintaining MT adjacencies

With the extension of IPv6 IS-IS MT, the new type, length, and value (TLV) parameters are added into the IS to IS hello (IIH) packets that advertise the topologies of the interface. In IPv6 IS-IS MT, the router advertises its information using the new TLV parameters such as MT ID TLV, MT IS Reachability TLV, MT

Reachable IPv4 TLV, and MT Reachable IPv6 TLV. The TLVs specify the types of data, the maximum length of the data, and the valid values for the data.

Forming adjacencies on the point-to-point interfaces

On a point-to-point interface, adjacencies are formed with IS-IS routers that do not implement MT extensions. If two peers share at least one common topology, then an adjacency is formed between the peers.

Forming adjacencies on the broadcast interfaces

On a broadcast interface, all the MT-enabled routers advertise their MT capability TLV in their IIH packets. The MT-enabled IS-IS routers form adjacency with any IS-IS routers whether or not MT is enabled. A peering MT-disabled IS-IS router does not form adjacency when NLPID TLVs do not match.

New TLV attributes

The new TLV parameters to support the IPv6 IS-IS MT extension are MT ID TLV, MT IS Reachability TLV, MT Reachable IPv4 TLV, and MT Reachable IPv6 TLV.

Enabling IPv6 IS-IS MT

You can enable IPv6 IS-IS MT in an area or a domain so that the MT-enabled devices runs IPv6 IS-IS in multi SPF mode. The following task enables IPv6 IS-IS MT with transition support.

Procedure

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

   ```
   device# configure terminal
   ```

2. Enter the **router isis** command to enter IS-IS router configuration mode and enable IS-IS on the device.

   ```
   device(config)# router isis
   ```

3. Enter the **address-family ipv6 unicast** command to enable ISIS address-family IPv6 unicast configuration mode.

   ```
   device(config-isis-router)# address-family ipv6 unicast
   ```

4. Enter the **multi-topology** command with the **transition** parameter to enable IPv6 IS-IS MT with transition support.

   ```
   device(config-router-isis-ipv6u)# multi-topology transition
   ```

Example

The following example enables IPv6 IS-IS MT with transition support.

```
device# configure terminal
device(config)# router isis
```

```
device(config-isis-router)# address-family ipv6 unicast
device(config-router-isis-ipv6u)# multi-topology transition
```

Configuration example to deploy IPv6 IS-IS MT

The figure below shows an example of a non-congruent topology enabled with IPv6 IS-IS MT. Device D1 supports both the IPv4 and IPv6 topologies, device D2 supports both the IPv4 and IPv6 topologies, device E2 supports an IPv4 topology, and device C2 supports both the IPv4 and IPv6 topologies.
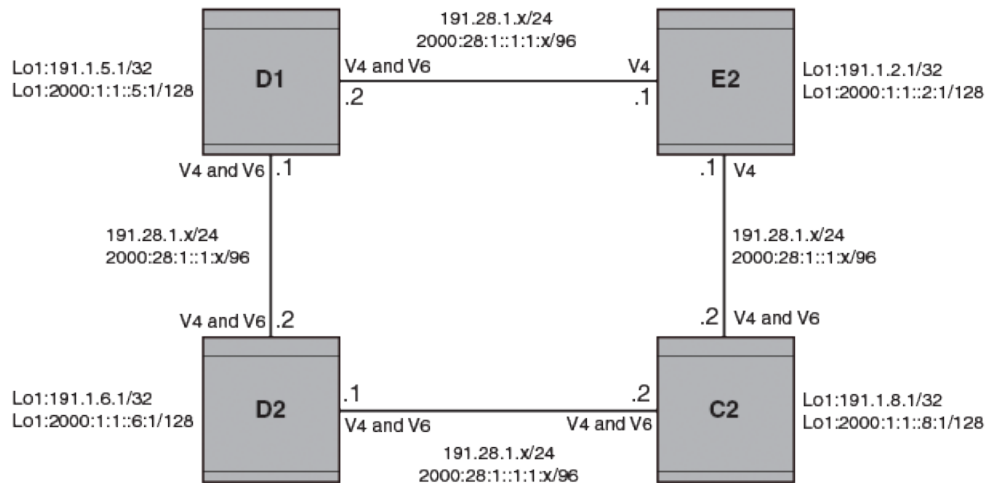
Figure 48: IPv6 IS-IS MT configuration

Configuration commands to enable IPv6 IS-IS MT on device D1

The following commands enable IPv6 IS-IS MT on device D1.

```
device# configure terminal
device(config)# router isis
device(config-isis-router)# net 00.0000.001b.ed03.1400.00
device(config-isis-router)# address-family ipv4 unicast
device(config-router-isis-ipv4u)# metric-style wide
device(config-router-isis-ipv4u)# exit
device(config-isis-router)# address-family ipv6 unicast
device(config-router-isis-ipv6u)# multi-topology
device(config-router-isis-ipv6u)# exit
```

Configuration commands to enable IPv6 IS-IS MT on device D2

The following commands enable IPv6 IS-IS MT on device D2.

```
device# configure terminal
device(config)# router isis
device(config-isis-router)# net 00.0000.001b.ed04.4400.00
device(config-isis-router)# address-family ipv4 unicast
device(config-router-isis-ipv4u)# metric-style wide
device(config-router-isis-ipv4u)# exit
device(config-isis-router)# address-family ipv6 unicast
device(config-router-isis-ipv6u)# multi-topology
device(config-router-isis-ipv6u)# exit
```

Configuration commands to enable IPv6 IS-IS MT on device E2

The following commands enable IPv6 IS-IS MT on device E2.

```
device# configure terminal
device(config)# router isis
device(config-isis-router)# net 00.0000.001b.ed04.4000.00
device(config-isis-router)# address-family ipv4 unicast
device(config-router-isis-ipv4u)# metric-style wide
device(config-router-isis-ipv4u)# exit
device(config-isis-router)# address-family ipv6 unicast
device(config-router-isis-ipv6u)# multi-topology
device(config-router-isis-ipv6u)# exit
```

Configuration commands to enable IPv6 IS-IS MT on device C2

The following commands enable IPv6 IS-IS MT on device C2.

```
device# configure terminal
device(config)# router isis
device(config-isis-router)# net 00.0000.001b.ed04.0000.00
device(config-isis-router)# address-family ipv4 unicast
device(config-router-isis-ipv4u)# metric-style wide
device(config-router-isis-ipv4u)# exit
device(config-isis-router)# address-family ipv6 unicast
device(config-router-isis-ipv6u)# multi-topology
device(config-router-isis-ipv6u)# exit
```

Displaying IS-IS statistics

Various **show isis** commands verify information about IPv4 and IPv6 IS-IS configurations.

Before You Begin

About This Task

Use one or more of the following commands to verify IS-IS information. The commands do not have to be entered in this order.

Procedure

1. Enter the **show isis** command.

```
device# show isis

IS-IS Routing Protocol Operation State: Enabled
IS-Type: Level-1-2
System Id: 768e.f805.5812
Manual area address(es): 11
Level-1-2 Database State: On
Administrative Distance 115
Maximum Paths 8
Default redistribution metric 0
Default link metric for level-1 0 (conf)/ 10 (adv)
Default link metric for level-2 0 (conf)/ 10 (adv)
Protocol Routes Redistributed into IS-IS: None
Number of Routes Redistributed into IS-IS: 0
Level-1 Auth-mode: None
Level-2 Auth-mode: None
```

```
Metric Style Supported for Level-1: Narrow
Metric Style Supported for Level-2: Narrow
Graceful-Restart Helper Support: Enabled
ISIS Partial SPF Optimizations: Enabled
Timers:
L1 SPF: Max-wait 5s Init-wait 5000ms Second-wait 5000ms
L2 SPF: Max-wait 5s Init-wait 5000ms Second-wait 5000ms
L1 SPF is not scheduled
L2 SPF is not scheduled
PSPF: Max-wait 5000ms Init-wait 2000ms Second-wait 5000ms
PSPF is not scheduled
LSP: max-lifetime 1200s refresh-interval 900s gen-interval 10s
retransmit-interval 5s, lsp-interval 33ms
SNP: csnp-interval 10s psnp-interval 2s
Global Hello Padding: Enabled
Global Hello Padding For Point to Point Circuits: Enabled
Ptpt Three Way HandShake Mechanism: Enabled
BGP Ipv4 Converged: False BGP Ipv6 Converged: False
IS-IS Traffic Engineering Support: Disabled
No ISIS Shortcuts Configured
BFD: Disabled, BFD HoldoverInterval: 0
NSR: Disabled
LSP-SYNC: Not Globally Enabled
```

This example output displays general IS-IS information..

2. Enter the **show isis config** command.

```
device# show isis config

router isis
net 11.768e.f805.5812.00
address-family ipv4 unicast
!
address-family ipv6 unicast
!
```

This example shows the global IS-IS configuration commands that are in effect on the device..

3. Enter the **show isis counts** command.

```
device# show isis counts

Area Mismatch: 0
Max Area Mismatch: 0
System ID Length Mismatch: 0
LSP Sequence Number Skipped: 0
LSP Max Sequence Number Exceeded: 0
Level-1 Database Overload: 1
Level-2 Database Overload: 0
Our LSP Purged: 2
```

This example shows IS-IS error statistics..

4. Enter the **show isis hostname** command.

```
device# show isis hostname

Total number of entries in IS-IS Hostname Table: 1
System ID Hostname * = local IS
---------------------------------------------------
* 768e.f805.5812 R1
```

This example shows the router-name-to-system-ID mapping table entries for the device.

5. Enter the **show isis interface** command.

```
device# show isis interface

Total number of IS-IS Interfaces: 11
Interface: Ve 301
Circuit State: UP Circuit Mode: Level 1-2
Circuit Type: BCAST Passive State: FALSE
Circuit Number: 2, MTU: 1500
Level-1 Auth-mode: NONE
Level-2 Auth-mode: NONE
Level-1 Metric: 10, Level-1 Priority: 64
Level-1 Hello Interval: 10, Level-1 Hello Multiplier: 3
Level-1 Designated IS: R1-02 Level-1 DIS Changes: 2
Level-2 Metric: 10, Level-2 Priority: 64
Level-2 Hello Interval: 10, Level-2 Hello Multiplier: 3
Level-2 Designated IS: R1-02 Level-2 DIS Changes: 2
Next IS-IS LAN Level-2 Hello in 11 seconds
Number of active Level-2 adjacencies: 0
Next IS-IS LAN Level-1 Hello in 1 seconds
Number of active Level-1 adjacencies: 0
Circuit State Changes: 1 Circuit Adjacencies State Changes: 0
Rejected Adjacencies: 0
Circuit Authentication L1 failures: 0
Circuit Authentication L2 failures: 0
Bad LSPs: 0
Control Messages Sent: 7577 Control Messages Received: 0
Hello Padding: Enabled
IP Enabled: TRUE
IP Addresses:
11.2.1.1/30
IPv6 Enabled: FALSE
MPLS TE Enabled: FALSE
BFD Enabled: FALSE
LDP-SYNC: Disabled, State:
…
```

This example shows information about IS-IS interfaces for a device.

6. Enter the **show isis interface** command, using the **brief** keyword.

```
device# show isis interface brief

Total number of IS-IS Interfaces: 11
Interface Type State Mode Passive MTU UpAdj DIS StateChg AdjStateChg
Ve 301 BCAST UP L12 FALSE 1500 0 None 1 0
Ve 302 BCAST UP L12 FALSE 1500 0 None 1 0
Ve 303 BCAST UP L12 FALSE 1500 0 None 1 0
Ve 304 BCAST UP L12 FALSE 1500 0 None 1 0
Ve 305 BCAST UP L12 FALSE 1500 0 None 1 0
Ve 306 BCAST UP L12 FALSE 1500 0 None 1 0
Ve 307 BCAST UP L12 FALSE 1500 0 None 1 0
Ve 308 BCAST UP L12 FALSE 1500 0 None 1 0
Ve 309 BCAST UP L12 FALSE 1500 0 None 1 0
Ve 310 BCAST UP L12 FALSE 1500 0 None 1 0
Lo 1 BCAST UP L12 TRUE 0 0 None 1 0
```

This example shows summarized information about IS-IS interfaces for a device.

7. Enter the **show isis neighbor** command.

```
device# show isis neighbor
```

This example shows IS-IS neighbor information.

8. Enter the **show isis spf-log** command.

```
device# show isis spf-log

ISIS Level-1 SPF Log
When Duration Nodes Count Last-Trigger-LSP Trigger
10h34m13s 0ms 1 10 R1.00-00 Interface State Change
10h34m38s 0ms 1 2 R1.00-00 Interface Config Change
10h34m43s 0ms 1 18 R1.00-00 Interface Config Change
10h34m48s 0ms 1 5 R1.00-00 Interface State Change
ISIS Level-2 SPF Log
When Duration Nodes Count Last-Trigger-LSP Trigger
10h34m13s 0ms 1 10 R1.00-00 Interface State Change
10h34m38s 0ms 1 2 R1.00-00 Interface Config Change
10h34m43s 0ms 1 18 R1.00-00 Interface Config Change
10h34m48s 0ms 1 5 R1.00-00 Interface State Change
```

This example shows IS-IS link-state packet (LSP) logging information.

9. Enter the **show isis traffic** command.

```
show isis traffic

                          Message Received      Message Sent
Level-1 Hellos            0                     44912
Level-2 Hellos            0                     44927
PTP Hellos               0                     0
Level-1 LSP              0                     0
Level-2 LSP              0                     0
Level-1 CSNP             0                     0
Level-2 CSNP             0                     0
Level-1 PSNP             0                     0
Level-2 PSNP             0                     0
```

This example shows information about IS-IS packet counts..

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.
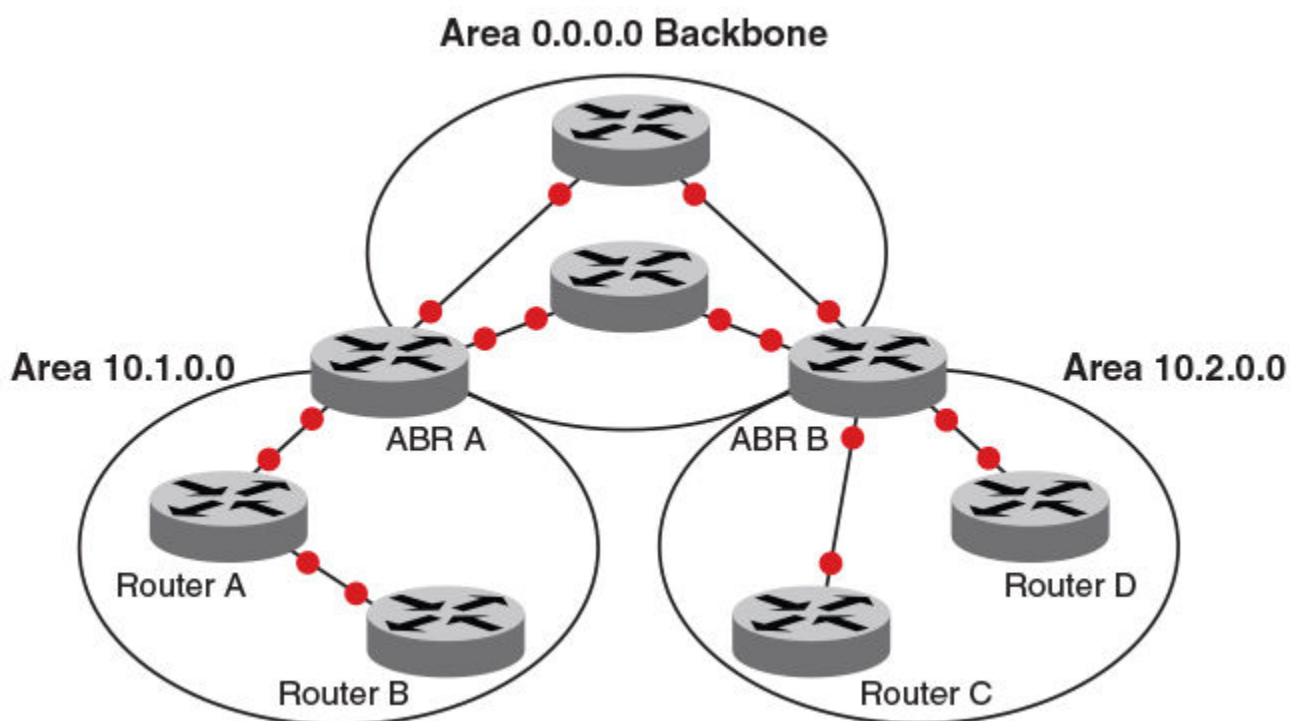
Figure 49: OSPF operating in a network

> **Note**
> For details of components and virtual links, refer to OSPFv2 components and roles on page 386 and Virtual links on page 395, 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 *Command Reference*.
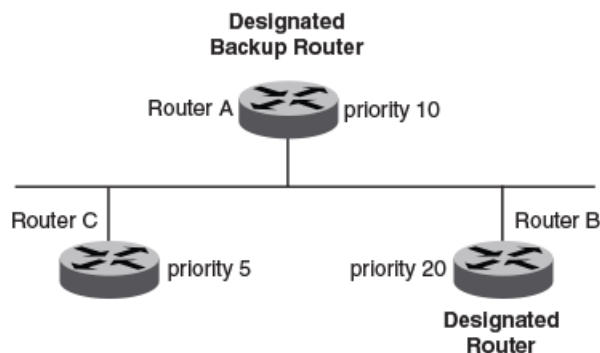
Figure 50: Designated and backup router election

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.

About This Task

Before You Begin

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.

Procedure

1. Enter the **router ospf** command in global 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 410.

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.

Assigning OSPFv2 areas

Areas can be assigned as OSPFv2 areas.

Before You Begin

About This Task

Procedure

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

   ```
   device# configure terminal
   ```
2. Enter the **router ospf** command to enter OSPF router configuration mode and enable OSPFv2 on the device.

   ```
   device(config)# router ospf
   ```
3. Enter the **area** command to define an OSPFv2 area ID.

   ```
   device(config-router-ospf-vrf-default-vrf)# area 0
   ```
4. Enter the **area** command to define a second OSPFv2 area ID.

   ```
   device(config-router-ospf-vrf-default-vrf)# area 10.1.1.1
   ```

Example

The following example assigns an OSPFv2 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)# router ospf
device(config-router-ospf-vrf-default-vrf)# area 0
device(config-router-ospf-vrf-default-vrf)# area 10.1.1.1
```

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.

About This Task

Procedure

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

    ```
    device# configure terminal
    ```

2. Enter the **router ospf** command to enter OSPF router configuration mode and enable OSPFv2 on the device.

    ```
    device(config)# router ospf
    ```

3. Enter the **area range** command, specifying an area ID, and enter the range. Use the **cost** parameter to specify the cost value for the area range. Repeat as necessary.

    ```
    device(config-ospf-router)# area 10.0.0.10 range 10.45.0.0 10.255.0.0 cost 20
    device(config-ospf-router)# area 10.0.0.20 range 10.45.0.0 10.255.0.0 cost 20
    ```

Example

The following example defines an area range for subnets on 10.0.0.10 and 10.0.0.20 and sets a cost of 20 for the area range.

```
device# configure terminal
device(config)# router ospf
device(config-ospf-router)# area 10.0.0.10 range 10.45.0.0 10.255.0.0 cost 20
device(config-ospf-router)# area 10.0.0.20 range 10.45.0.0 10.255.0.0 cost 20
```

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.

About This Task

Procedure

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

   ```
   device# configure terminal
   ```

2. Enter the **router ospf** command to enter OSPF router configuration mode and enable OSPFv2 on the device.

   ```
   device(config)# router ospf
   ```

3. 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
   ```

Example

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)# 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.
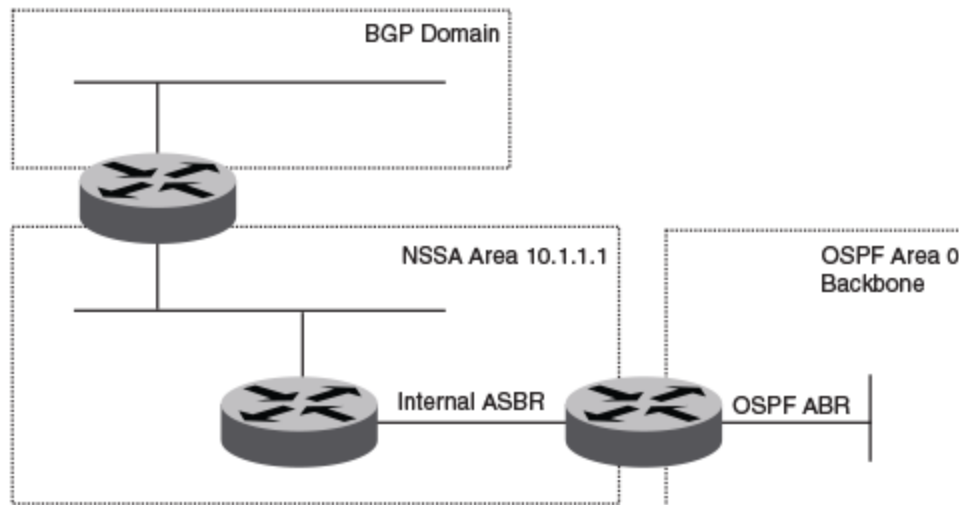
Figure 51: OSPF network containing an NSSA

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.

Configuring an NSSA

About This Task

OSPFv2 areas can be defined as NSSA areas with modifiable parameters.

Procedure

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

   ```
   device# configure terminal
   ```

2. Enter the **router ospf** command to enter OSPF router configuration mode and enable OSPFv2 on the device.

   ```
   device(config)# router ospf
   ```

3. 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.

Example

The following example configures OSPF area 10.1.1.1 as an NSSA.

```
device# configure terminal
device(config)# router ospf
device(config-router-ospf-vrf-default-vrf)# area 10.1.1.1 nssa 1
```

Configuring a summary-address for the NSSA

About This Task

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.

Procedure

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

   ```
   device# configure terminal
   ```

2. Enter the **router ospf** command to enter OSPF router configuration mode and enable OSPFv2 on the device.

   ```
   device(config)# router ospf
   ```

3. 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
   ```

4. 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
   ```

Example

The following example configures a summary-address in NSSA 10.1.1.1.

```
device# configure terminal
device(config)# 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
```

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 OSPFv2 device. When a port is assigned to an area, all corresponding subnets on that port are automatically included in the assignment.

About This Task

To assign a loopback interface to an area with the IP address of 10.5.0.0, perform the following task:

Procedure

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 loopback 2
   ```

3. Enter the **ip ospf area** command followed by the IP address of the area.

   ```
   device(config-Loopback-2)# ip ospf area 10.5.0.0
   ```

Example

The following example assigns a loopback interface to an area with the IP address of 10.5.0.0.

```
device# configure terminal
device(config)# interface loopback 2
device(config-Loopback-2)# 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.

Configuring an MD5 password and authentication change hold time for an OSPFv2 interface

An MD5 password and authentication change hold time can be configured. This task sets an MD5 password and sets an authentication change hold time for an OSPFv2 Ethernet interface

About This Task

Procedure

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 ethernet 1/1
   ```

3. Enter the **ip ospf md5-authentication** command with the **key-activation-wait-time** parameter, and specify a time interval, to set the time that OSPFv2 waits before activating a new MD5 key.

   ```
   device(conf-if-eth-1/1)# ip ospf md5-authentication key-activation-wait-time 240
   ```

4. Enter the **ip ospf md5-authentication** command with the **key-id** and **key** parameters to set the MD5 key ID and a password.

   ```
   device(conf-if-eth-1/1)# ip ospf md5-authentication key-id 22 key myospfpassword
   ```

Example

The following example sets the time that OSPFv2 waits before activating a new MD5 key to 240 seconds on an Ethernet interface, and sets the MD5 key ID to 22 and a password "myospfpassword" on an Ethernet interface.

```
device# configure terminal
device(config)# interface ethernet 1/1
device(conf-if-eth-1/1)# ip ospf md5-authentication key-activation-wait-time 240
device(conf-if-eth-1/1)# ip ospf md5-authentication key-id 22 key myospfpassword
```

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.
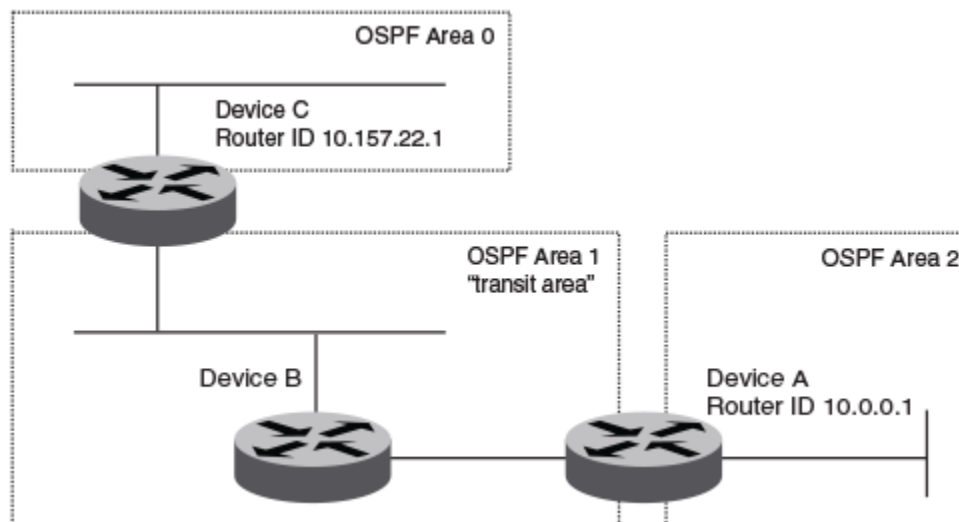
Figure 52: Defining OSPF virtual links within a network

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.

Before You Begin

About This Task

A virtual link is configured, and a virtual link endpoint on two devices, ABR1 and ABR2, is defined.

Procedure

1. On ABR1, enter the **configure terminal** command to access global configuration mode.

```
device# configure terminal
```

2. Enter the **router ospf** command to enter OSPF router configuration mode and enable OSPFv2 on the device.

```
device(config)# router ospf
```

3. Enter the **area** command to assign an OSPFv2 area ID.

```
device(config-router-ospf-vrf-default-vrf)# area 0
```

4. Enter the **area** command to assign an OSPFv2 area ID.

```
device(config-router-ospf-vrf-default-vrf)# area 1
```

5. 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
```

6. On ABR2, enter the **configure terminal** command to access global configuration mode.

```
device# configure terminal
```

7. Enter the **router ospf** command to enter OSPFv2 router configuration mode and enable OSPFv2 on the device.

```
device(config)# router ospf
```

8. Enter the **area** command to assign an OSPFv2 area ID.

```
device(config-router-ospf-vrf-default-vrf)# area 1
```

9. Enter the **area** command to assign an OSPFv2 area ID.

```
device(config-router-ospf-vrf-default-vrf)# area 2
```

10. 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
```

Example

The following example configures a virtual link between two devices.

```
ABR1:
device1# configure terminal
device1(config)# 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)# 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.

Before You Begin

About This Task

Procedure

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

    ```
    device# configure terminal
    ```

2. Enter the **router ospf** command to enter OSPF router configuration mode and enable OSPFv2 globally.

    ```
    device(config)# router ospf
    ```

3. 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
    ```

Example

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)# 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.

About This Task

Procedure

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

   ```
   device# configure terminal
   ```

2. Enter the **router ospf** command to enter OSPF router configuration mode and enable OSPFv2 globally.

   ```
   device(config)# router ospf
   ```

3. 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).

Example

The following example changes the OSPFv2 LSA pacing interval is changed to 120 seconds (2 minutes).

```
device# configure terminal
device(config)# 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 by 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 that 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 can quickly resume full operation because of 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**
> - 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.
> - Process restart takes precedence over GR or non-stop routing (NSR).

Disabling OSPFv2 graceful restart

OSPFv2 graceful restart (GR) is enabled by default, and can be disabled on a routing device.

Before You Begin

About This Task

Procedure

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

   ```
   device# configure terminal
   ```

2. Enter the **router ospf** command to enter OSPF router configuration mode and enable OSPFv2 globally.

   ```
   device(config)# router ospf
   ```

3. Enter the **no graceful restart** command to disable GR on the device.

   ```
   device(config-router-ospf-vrf-default-vrf)# no graceful-restart
   ```

Example

The following example disables GR.

```
device# configure terminal
device(config)# 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.

Before You Begin

About This Task

> **Note**
> GR is mutually exclusive to NSR.

Procedure

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

   ```
   device# configure terminal
   ```

2. Enter the **router ospf** command to enter OSPF router configuration mode and enable OSPFv2 globally.

   ```
   device(config)# router ospf
   ```

3. Enter the **graceful restart** command to re-enable GR on the device.

```
device(config-router-ospf-vrf-default-vrf)# graceful-restart
```

4. 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
```

Example

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)# 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.

Before You Begin

About This Task

Procedure

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

```
device# configure terminal
```

2. Enter the **router ospf** command to enter OSPF router configuration mode and enable OSPFv2 on the device.

```
device(config)# router ospf
```

3. 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
```

Example

The following example disables the GR helper.

```
device# configure terminal
device(config)# 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 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 virtual links, so traffic loss is expected during hitless failover.
> - NSR and GR are mutually exclusive.
> - Process restart takes precedence over GR and NSR.

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.

Before You Begin

About This Task

> **Note**
> GR is mutually exclusive to NSR.

Procedure

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

   ```
   device# configure terminal
   ```

2. Enter the **router ospf** command to enter OSPF router configuration mode and enable OSPFv2 globally.

   ```
   device(config)# router ospf
   ```

3. Enter the **nonstop-routing** command to re-enable NSR on the device.

   ```
   device(config-router-ospf-vrf-default-vrf)# nonstop-routing
   ```

Example

The following example re-enables NSR for OSPFv2.

```
device# configure terminal
device(config)# router ospf
device(config-router-ospf-vrf-default-vrf)# nonstop-routing
```

Redistributing routes into OSPFv2

OSPFv2 routes can be redistributed, and the routes to be redistributed can be specified.

About This Task

The redistribution of BGP, connected, and static IP routes into OSPFv2 is configured on a device.

Procedure

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

   ```
   device# configure terminal
   ```

2. Enter the **router ospf** command to enter OSPFv2 router configuration mode and enable OSPFv2 on the device.

   ```
   device(config)# router ospf
   ```

3. Enter the **redistribute** command with the **static** parameter to redistribute static routes.

   ```
   device(config-router-ospf-vrf-default-vrf)# redistribute static
   ```

4. Enter the **redistribute** command with the **connected** parameter to redistribute static routes.

   ```
   device(config-router-ospf-vrf-default-vrf)# redistribute connected
   ```

5. Enter the **redistribute** command with the **bgp** parameter to redistribute BGP routes.

   ```
   device(config-router-ospf-vrf-default-vrf)# redistribute bgp
   ```

Example

The following example redistributes static and BGP routes into OSPFv2 on a device.

```
device# configure terminal
device(config)# router ospf
device(config-router-ospf-vrf-default-vrf)# redistribute static
device(config-router-ospf-vrf-default-vrf)# redistribute connected
device(config-router-ospf-vrf-default-vrf)# redistribute bgp
```

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.
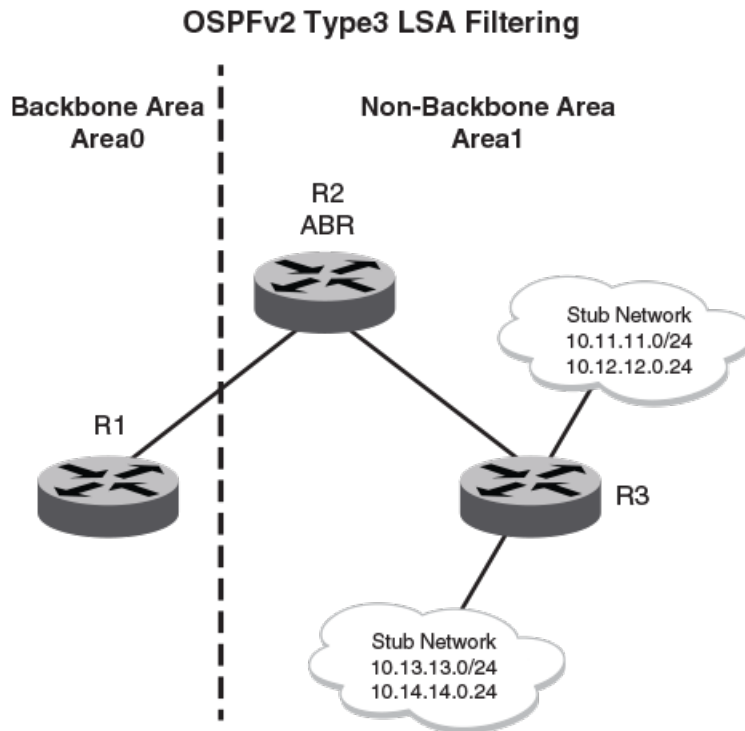
Figure 53: OSPFv2 type 3 LSA filtering

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.

Table 37: Behavior for prefix list configurations

IP prefix list	OSPF area prefix list	Event	Filtering done
XXX	Not defined	None	No (permit all)
Not defined	Defined	None	Yes (deny all)
Not defined	Defined	IP prefix list defined	Recalculation
Defined (no rules configured)	Defined	None	Implicit deny (deny all)
Defined (rules configured)	Defined	IP prefix list deleted	Recalculation and deny all
Defined (rules configured)	Defined	IP prefix list rule added or modified or deleted	Recalculation
Defined (rules configured)	Defined	Area prefix list deleted	Recalculation and permit all

Configuring OSPFv2 type 3 LSA filtering

About This Task

An IP prefix list can be configured and applied to an OSPFv2 area.

Procedure

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

   ```
   device# configure terminal
   ```

2. Enter the **ip prefix-list** command and specify a name.

   ```
   device(config)# ip prefix-list mylist permit 10.1.0.0/16
   ```
   Configures an IP prefix list named "mylist" that permits routes to network 10.1.0.0/16.

3. Enter the **router ospf** command to enter OSPF router configuration mode and enable OSPFv2 on the device.

   ```
   device(config)# router ospf
   ```

4. Enter the **area** command to define an OSPFv2 area ID.

   ```
   device(config-router-ospf-vrf-default-vrf)# area 1
   ```

5. Enter the **area prefix-list** command and specify an area address and a prefix-list.

   ```
   device(config-router-ospf-vrf-default-vrf)# area 1 prefix-list mylist out
   ```
   Configures the device to use the IP prefix list "mylist" to determine which routes from Area 1 are advertised to other areas. The device advertises routes that fall within the 10.1.0.0/16 range to other areas because the IP prefix list explicitly permits these routes to be sent to other areas from Area 1.

Example

The following example onfigures an IP prefix list named "mylist" that permits routes to network 10.1.0.0/16. This list is used to determine which routes to send to Area 1.

```
device# configure terminal
device(config)# ip prefix-list mylist permit 10.1.0.0/16
device(config)# router ospf
device(config-router-ospf-vrf-default-vrf)# area 1
device(config-router-ospf-vrf-default-vrf)# area 1 prefix-list mylist out
```

OSPFv2 over VRF

OSPFv2 can run over multiple Virtual Routing and Forwarding (VRF) instances. All OSPFv2 commands 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.

About This Task

Before You Begin
A non-default VRF instance has been configured.

Procedure

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

   ```
   device# configure terminal
   ```

2. 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)# router ospf vrf green
   ```

Example

The following example enables OSPFv2 in a non-default VRF.

```
device# configure terminal
device(config)# router ospf vrf green
device(config-router-ospf-vrf-green)#
```

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.

About This Task

> **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.

Procedure

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

   ```
   device# configure terminal
   ```

2. Enter the **router ospf** command to enter OSPF router configuration mode and enable OSPFv2 on the device.

   ```
   device(config)# router ospf
   ```

3. 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
   ```

Example

The following example configures an OSPFv2 device to advertise the maximum metric value using the **all-lsas** parameter.

```
device# configure terminal
device(config)# 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.

Before You Begin

Procedure

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

   ```
   device# configure terminal
   ```

2. Enter the **ip router ospf** command to enter OSPF router configuration mode and enable OSPFv2 on the device.

```
device(config)# router ospf
```

3. Enter the **rfc1583-compatibility** command to re-enable OSPFv2 compatibility with RFC 1583.

```
device(config-router-ospf-vrf-default-vrf)# rfc1583-compatibility
```

Example

The following example re-enables OSPFv2 compatibility with RFC 1583.

```
device# configure terminal
device(config)# router ospf
device(config-router-ospf-vrf-default-vrf)# rfc1583-compatibility
```

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.

Before You Begin

Procedure

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

```
device# configure terminal
```

2. Enter the **router ospf** command to enter OSPF router configuration mode and enable OSPFv2 globally.

```
device(config)# router ospf
```

3. Enter the **no log all** command to disable the logging of all OSPFv2 events.

```
device(config-router-ospf-vrf-default-vrf)# no log all
```

Example

The following example re-enables the logging of all OSPFv2 events.

```
device# configure terminal
device(config)# 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.

Disabling OSPFv2

About This Task

To disable OSPFv2 on a device, use the **no router ospf** command:

Procedure

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

   ```
   device# configure terminal
   ```

2. Enter the **no router ospf** command to disable OSPFv2 on the device.

   ```
   device(config)# no router ospf
   ```

Example

The following example disables OSPFv2 on a device.

```
device# configure terminal
device(config)# no router ospf
```

Displaying OSPFv2 results

The **show ip ospf** command and its variations can be used to display information about OSPFv2 configurations.

Before You Begin

About This Task

Use one or more of the following commands to verify OSPFv2 information. Using the **show ip ospf** command is optional, and the variations of the command can be entered in any order.

Procedure

1. In privileged EXEC mode, enter the **show ip ospf** command to display general OSPFv2 information.

```
device# show ip ospf

show ip ospf
OSPF Version                  Version 2
Router Id                     32.32.32.32
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     0
Originate New LSA Counter     9
Rx New LSA Counter            5
External LSA Limit            14913080
Administrative Distance
  -  External Routes:         110
  -  Intra Area Routes:       110
  -  Inter Area Routes:       110
Database Overflow Interval    0
Database Overflow State :     NOT OVERFLOWED
RFC 1583 Compatibility :      Disabled
NSSA Translator:              Enabled
Nonstop Routing:              Disabled
Graceful Restart              Enabled
Graceful Restart Helper       Enabled
Graceful Restart Time         120
LDP-SYNC: Not globally enabled
Interfaces with LDP-SYNC enabled:
    None
```

The example output displays general OSPFv2 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.

2. Enter the **show ip ospf area** command to display information about an OSPFv2 area.

```
device# show ip ospf area

Number of Areas is 1

Index   Area             Type  Cost      SPFR      ABR   ASBR  LSA   Chksum(Hex)
1       0                normal 0        482       0     0     1     00006828
```

The example output displays detailed output for OSPFv2 Area 0.

3. Enter the **show ip ospf interface** command to display OSPFv2 interface information.

```
device# show ip ospf interface

Ethernet 1/8 admin up, oper up
    IP Address 10.1.8.32, Area 0
```

```
     BFD is disabled
     Database Filter: Not Configured
     State DR, Pri 1, Cost 1, Options ------E-, Type broadcast Events 8
     Timers(sec): Transmit 1, Retrans 5, Hello 10, Dead 40
     DR:  Router ID 32.32.32.32       Interface Address 10.1.8.32
     BDR: Router ID 0.0.0.0           Interface Address 0.0.0.0
     Neighbor Count = 1, Adjacent Neighbor Count= 0
     Neighbor:         10.1.8.19 [id 19.19.19.18]
     Authentication-Key: None
     MD5 Authentication: Key None, Key-Id  None , Auth-change-wait-time 300
LDP-SYNC: Disabled, State: -

Loopback 1 admin up, oper up
     IP Address 32.32.32.32, Area 0
     BFD is disabled
     Database Filter: Not Configured
     State DR, Pri 1, Cost 1, Options ------E-, Type broadcast Events 2
     Timers(sec): Transmit 1, Retrans 5, Hello 10, Dead 40
     DR:  Router ID 32.32.32.32       Interface Address 32.32.32.32
     BDR: Router ID 0.0.0.0           Interface Address 0.0.0.0
     Neighbor Count = 0, Adjacent Neighbor Count= 0
     Authentication-Key: None
     MD5 Authentication: Key None, Key-Id  None , Auth-change-wait-time 300
LDP-SYNC: Disabled, State: -
```

The example output displays detailed output for OSPFv2 interfaces.

4. Enter the **show ip ospf interface** command with the **brief** parameter to display summarized OSPFv2 interface information.

```
device# show ip ospf interface brief
Interface       Area              IP Addr/Mask         Cost  State    Nbrs(F/C)
Eth 6/45        0                 10.10.10.1/24        1     down     0/0
Ve 15           0                 15.15.15.1/24        1     DR       0/0
Ve 161          0                 161.161.161.1/24     1     DR       0/0
Ve 162          0                 162.162.162.1/24     1     DR       0/0
```

The example output displays summarized output for OSPFv2 interfaces.

5. Enter the **show ip ospf neighbor** command to display OSPFv2 neighbor information for the device.

```
device# show ip ospf neighbor

Number of Neighbors is 1, in FULL state 0

Port            Address         Pri State      Neigh Address   Neigh ID        Ev
Opt Cnt
Eth 1/8         10.1.8.32       1   INIT/OTHER 10.1.8.19       19.19.19.18     7
66  0
```

The example output displays output for an OSPFv2 neighbor.

6. Enter the **show ip ospf routes** command to display information about OSPFv2 routes.

```
device# show ip ospf routes

OSPF Regular Routes 7:

    Destination         Mask             Path_Cost Type2_Cost Path_Type
    1.1.1.1             255.255.255.255 1         0          Intra
    Adv_Router          Link_State       Dest_Type State      Tag       Flags
    1.1.1.1             1.1.1.1          Network   Valid      0         6
```

```
       Paths Out_Port      Next_Hop        Type       State
       1     Lo 1          0.0.0.0         OSPF       0  0

       Destination         Mask            Path_Cost Type2_Cost Path_Type
       1.1.1.2             255.255.255.255 1         0          Intra
       Adv_Router          Link_State      Dest_Type State      Tag       Flags
       1.1.1.1             1.1.1.1         Network   Valid      0          6
       Paths Out_Port      Next_Hop        Type       State
       1     Lo 2          0.0.0.0         OSPF       0  0

       Destination         Mask            Path_Cost Type2_Cost Path_Type
       1.1.1.3             255.255.255.255 1         0          Intra
       Adv_Router          Link_State      Dest_Type State      Tag       Flags
       1.1.1.1             1.1.1.1         Network   Valid      0          6
       Paths Out_Port      Next_Hop        Type       State
       1     Lo 3          0.0.0.0         OSPF       0  0

       Destination         Mask            Path_Cost Type2_Cost Path_Type
       1.1.1.4             255.255.255.255 1         0          Intra
       Adv_Router          Link_State      Dest_Type State      Tag       Flags
       1.1.1.1             1.1.1.1         Network   Valid      0          6
       Paths Out_Port      Next_Hop        Type       State
       1     Lo 4          0.0.0.0         OSPF       0  0

       Destination         Mask            Path_Cost Type2_Cost Path_Type
       1.1.1.5             255.255.255.255 1         0          Intra
       Adv_Router          Link_State      Dest_Type State      Tag       Flags
       1.1.1.1             1.1.1.1         Network   Valid      0          6
       Paths Out_Port      Next_Hop        Type       State
       1     Lo 5          0.0.0.0         OSPF       0  0

       Destination         Mask            Path_Cost Type2_Cost Path_Type
       1.1.1.6             255.255.255.255 1         0          Intra
       Adv_Router          Link_State      Dest_Type State      Tag       Flags
       1.1.1.1             1.1.1.1         Network   Valid      0          6
       Paths Out_Port      Next_Hop        Type       State
       1     Lo 6          0.0.0.0         OSPF       0  0

       Destination         Mask            Path_Cost Type2_Cost Path_Type
       1.1.1.7             255.255.255.255 1         0          Intra
       Adv_Router          Link_State      Dest_Type State      Tag       Flags
       1.1.1.1             1.1.1.1         Network   Valid      0          6
       Paths Out_Port      Next_Hop        Type       State
       1     Lo 7          0.0.0.0         OSPF       0  0
```

7. Enter the **show ip ospf summary** command to display summary output for OSPFv2 sessions.

```
device# show ip ospf summary

Seq Instance         Intfs   Nbrs    Nbrs-Full LSAs    Routes
1   default-vrf      2       1       0         1       2
```

8. Enter the **show ip ospf summary all-vrfs total** command to display cumulative summary output for OSPFv2 sessions.

```
device# show ip ospf summary all-vrfs total
--------------------------------------------
       IPv4 OSPF VRFs Summary Total
--------------------------------------------
Number of VRFs: 1
Number of Interfaces: 200
Number of Neighbors: 200
```

```
            Number of Neighbors in Full state: 200
            Number of LSAs: 182600
            Number of Routes: 102600
```

9. Enter the **show ip ospf traffic** command to display OSPF traffic details.

```
device# show ip ospf traffic

                Packets Received        Packets Sent
Hello                      3943                 3936
Database                      6                    6
LSA Req                       1                    2
LSA Upd                      18                   20
LSA Ack                      17                   16
No Packet Errors!
```

Supported scale for OSPFv2 and performance considerations

Scalability of OSPFv2 gets limited by availability of resources like Memory, CPU and hardware tables. While increasing the scale, you need to consider time taken for adjacency formation and route convergence.

The supported scale for OSPFv2 across all VRF instances is as follows:

Table 38: Supported scale across all VRFs

Description	Number supported
Maximum number of OSPFv2 VRF Instances	1024
Maximum number of OSPFv2 Areas	1024
Number of OSPFv2 Interfaces	2048
Number of OSPFv2 Neighbors	2048
Maximum Number of OSPFv2 Routes	100k (102400)

The supported scale for OSPFv2 per VRF instance is as follows:

Table 39: Supported scale per VRF

Description	Number supported
Maximum number of OSPFv2 Areas	200
Number of OSPFv2 Interfaces	200
Number of OSPFv2 Neighbors	200
Maximum Number of OSPFv2 Routes	100k (102400)

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.

Configuring the router ID

When configuring OSPFv3, the router ID for a device must be specified.

Before You Begin

Procedure

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

   ```
   device# configure terminal
   ```

2. Enter the **ip router-id** command to specify the router ID.

   ```
   device(config) ip router-id 10.11.12.13
   ```

Example

The following example configures the router ID for a device.

```
device# configure terminal
device(config)# 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.

About This Task

Procedure

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

   ```
   device# configure terminal
   ```

2. Enter the `ip router-id` command to specify the router ID.

```
device(config) ip router-id 10.11.12.13
```

3. Enter the `ipv6 router ospf` command to enter OSPFv3 router configuration mode and enable OSPFv3 on the device.

```
device(config)# ipv6 router ospf
```

Example

The following example enables OSPFv3 on a device.

```
device# configure terminal
device(config)# ip router-id 10.11.12.13
device(config)# 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.

Before You Begin

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.

About This Task

Procedure

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

   ```
   device# configure terminal
   ```

2. Enter the **ip router-id** command to specify the router ID.

   ```
   device(config) ip router-id 10.11.12.13
   ```

3. Enter the **ipv6 router ospf** command to enter OSPFv3 router configuration mode and enable OSPFv3 on the device.

   ```
   device(config)# ipv6 router ospf
   ```

4. Enter the **area** command to define an OSPFv3 area ID.

   ```
   device(config-ipv6-router-ospf-vrf-default-vrf)# area 0
   ```

5. Enter the **area** command to define a second OSPFv3 area ID.

   ```
   device(config-ipv6-router-ospf-vrf-default-vrf)# area 10.1.1.1
   ```

Example

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)# ip router-id 10.11.12.13
device(config)# 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.

Before You Begin

Ensure that OSPFv3 areas are assigned.

About This Task

> **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.

Procedure

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 ethernet 1/1
   ```

3. Enter the **ipv6 address** command to add an IPv6 address to the interface.

   ```
   device(conf-if-eth-1/1)# ipv6 address 2001:1:0::1:1/64
   ```

4. Enter the **ipv6 ospf area** command.

   ```
   device(conf-if-eth-1/1)# 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(conf-if-eth-1/1)# exit
   ```

6. Enter the **interface** command and specify an interface.

   ```
   device(config)# interface ethernet 2/1
   ```

7. Enter the **ipv6 address** command to add an IPv6 address to the interface.

   ```
   device(conf-if-eth-2/1)# ipv6 address 2001:1:0::2:1/64
   ```

8. Enter the **ipv6 ospf area** command.

   ```
   device(conf-if-eth-2/1)# ipv6 ospf area 1
   ```
   Area 1 is assigned to the specified interface with the IPv6 address of 2001:1:0:2::1/64.

Example

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 ethernet 1/1
device(conf-if-eth-1/1)# ipv6 address 2001:1:0::1:1/64
device(conf-if-eth-1/1)# ipv6 ospf area 0
device(conf-if-eth-1/1)# exit
device(config)# interface ethernet 2/1
device(conf-if-eth-2/1)# ipv6 address 2001:1:0::2:1/64
device(conf-if-eth-2/1)# 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.

About This Task

Procedure

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

   ```
   device# configure terminal
   ```

2. Enter the **ip router-id** command to specify the router ID.

   ```
   device(config) ip router-id 10.4.4.4
   ```

3. Enter the **ipv6 router ospf** command to enter OSPFv3 router configuration mode and enable OSPFv3 on the device.

   ```
   device(config)# ipv6 router ospf
   ```

4. 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.

Example

The following example sets an additional cost of 100 on a stub area defined as 4.

```
device# configure terminal
device(config)# ip router-id 10.4.4.4
device(config)# 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.

About This Task

Procedure

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

   ```
   device# configure terminal
   ```

2. Enter the **ip router-id** command to specify the router ID.

   ```
   device(config) ip router-id 10.3.3.3
   ```

3. Enter the **ipv6 router ospf** command to enter OSPFv3 router configuration mode and enable OSPFv3 on the device.

   ```
   device(config)# ipv6 router ospf
   ```

4. 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.

Example

The following example sets an additional cost of 33 on an NSSA defined as 3.

```
device# configure terminal
device(config)# ip router-id 10.3.3.3
device(config)# 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.
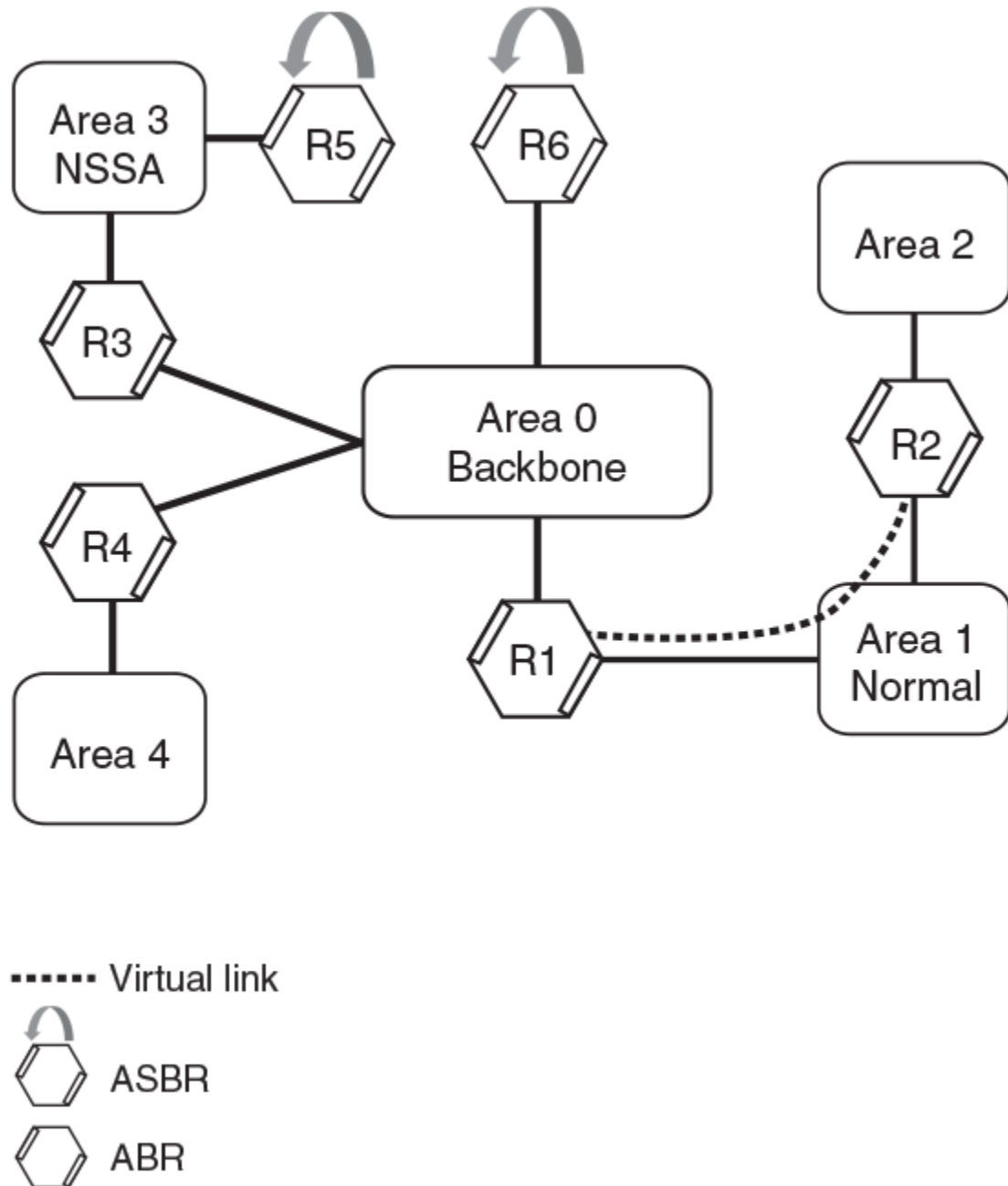
Figure 54: OSPFv3 virtual link

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.

Before You Begin

About This Task

A virtual link is configured, and a virtual link endpoint on two devices, ABR1 and ABR2, is defined.

Procedure

1. On ABR1, enter the **configure terminal** command to access global configuration mode.

   ```
   device# configure terminal
   ```

2. Enter the **ip router-id** command to specify the router ID.

   ```
   device(config) ip router-id 10.1.1.1
   ```

3. Enter the **ipv6 router ospf** command to enter OSPFv3 router configuration mode and enable OSPFv3 on the device.

```
device(config)# ipv6 router ospf
```

4. Enter the **area** command to assign an OSPFv3 area ID.

```
device(config-ipv6-router-ospf-vrf-default-vrf)# area 0
```

5. Enter the **area** command to assign an OSPFv3 area ID.

```
device(config-ipv6-router-ospf-vrf-default-vrf)# area 1
```

6. 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
```

7. On ABR2, enter the **configure terminal** command to access global configuration mode.

```
device# configure terminal
```

8. Enter the **ip router-id** command to specify the router ID.

```
device(config) ip router-id 10.2.2.2
```

9. Enter the **ipv6 router ospf** command to enter OSPFv3 router configuration mode and enable OSPFv3 on the device.

```
device(config)# ipv6 router ospf
```

10. Enter the **area** command to assign an OSPFv3 area ID.

```
device(config-ipv6-router-ospf-vrf-default-vrf)# area 1
```

11. Enter the **area** command to assign an OSPFv3 area ID.

```
device(config-ipv6-router-ospf-vrf-default-vrf)# area 2
```

12. 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
```

Example

The following example configures a virtual link between two devices.

```
ABR1:
device1# configure terminal
device1(config)# ip router-id 10.1.1.1
device1(config)# 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)# ip router-id 10.2.2.2
device2(config)# 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+
- IPv6 IS-IS

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.

About This Task

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.

Procedure

1. On device1, enter the **configure terminal** command to access global configuration mode.

```
device# configure terminal
```

2. Enter the **ipv6 router ospf** command to enter OSPFv3 router configuration mode and enable OSPFv3 on the device.

```
device(config)# ipv6 router ospf
```

3. Enter the **redistribute** command with the **static** parameter to redistribute static routes.

```
device(config-ipv6-router-ospf-vrf-default-vrf)# redistribute static
```

4. Enter the **redistribute** command with the **bgp** parameter to redistribute static routes.

```
device(config-ipv6-router-ospf-vrf-default-vrf)# redistribute bgp
```

5. On device2, enter the **configure terminal** command to access global configuration mode.

```
device# configure terminal
```

6. Enter the **ipv6 router ospf** command to enter OSPFv3 router configuration mode and enable OSPFv3 on the device.

```
device(config)# ipv6 router ospf
```

7. 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
```

Example

The following example redistributes static and BGP routes routes into OSPFv3 on a device.

```
device# configure terminal
device(config)# ipv6 router ospf
device(config-ipv6-router-ospf-vrf-default-vrf)# redistribute static
device(config-ipv6-router-ospf-vrf-default-vrf)# redistribute bgp
```

Example

The following example redistributes connected routes into OSPFv3 on a device and specifies a route map.

```
device# configure terminal
device(config)# 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.

About This Task

Procedure

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

   ```
   device# configure terminal
   ```

2. Enter the **ipv6 router ospf** command to enter OSPFv3 router configuration mode and enable OSPFv3 globally.

   ```
   device(config)# ipv6 router ospf
   ```

3. 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.

Example

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)# 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.

Before You Begin

Procedure

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

   ```
   device# configure terminal
   ```

2. Enter the **ipv6 router ospf** command to enter OSPFv3 configuration mode and enable OSPFv3 globally.

   ```
   device(config)# ipv6 router ospf
   ```

3. Enter the **no log** command to disable the logging of OSPFv3 events.

   ```
   device(config-ipv6-router-ospf-vrf-default-vrf)# no log
   ```

Example

The following example re-enables the logging of OSPFv3 events.

```
device# configure terminal
device(config)# 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 0 seconds. You can configure the SPF delay to a value from 0 through 65535 seconds. When the SPF delay is set 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 0 seconds. You can configure the SPF hold time to a value from 0 through 65535 seconds. When the SPF hold time is set 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.

Before You Begin

Procedure

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

   ```
   device# configure terminal
   ```

2. Enter the **ipv6 router ospf** command to enter OSPFv3 router configuration mode and enable OSPFv3 globally.

   ```
   device(config)# ipv6 router ospf
   ```

3. 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.

Example

The following example changes the SPF delay and hold time.

```
device# configure terminal
device(config)# 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.

Before You Begin

Procedure

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

   ```
   device# configure terminal
   ```

2. Enter the **ipv6 router ospf** command to enter OSPFv3 router configuration mode and enable OSPFv3 globally.

   ```
   device(config)# ipv6 router ospf
   ```

3. 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.

4. 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.

5. 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.

Example

The following example changes the default administrative distances for intra-area routes, inter-area routes, and external routes.

```
device# configure terminal
device(config)# 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.

Before You Begin

Procedure

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

   ```
   device# configure terminal
   ```

2. Enter the **ipv6 router ospf** command to enter OSPFv3 router configuration mode and enable OSPFv3 globally.

   ```
   device(config)# ipv6 router ospf
   ```

3. 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
   ```

Example

The following example changes the auto-cost reference bandwidth to 500.

```
device# configure terminal
device(config)# 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.

About This Task

Procedure

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

   ```
   device# configure terminal
   ```

2. Enter the **ipv6 router ospf** command to enter OSPFv3 router configuration mode and enable OSPFv3 globally.

   ```
   device(config)# ipv6 router ospf
   ```

3. 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).

Example

The following example restores the pacing interval to the default value of 240 seconds (4 minutes).

```
device# configure terminal
device(config)# 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 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 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.

Before You Begin
A non-default VRF instance has been configured.

Procedure

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

   ```
   device# configure terminal
   ```

2. Enter the **ipv6 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)# ipv6 router ospf vrf green
   ```

Example

The following enables OSPFv3 in a non-default VRF.

```
device# configure terminal
device(config)# ipv6 router ospf vrf green
device(config-ipv6-router-ospf-vrf-green)#
```

Setting all OSPFv3 interfaces to the passive state

All OSPFv3 interfaces can be set as passive, causing them to drop all OSPFv3 control packets.

Before You Begin

Procedure

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

    ```
    device# configure terminal
    ```

2. Enter the **ipv6 router ospf** command to enter OSPFv3 router configuration mode and enable OSPFv3 globally.

    ```
    device(config)# ipv6 router ospf
    ```

3. Enter the **default-passive-interface** command to mark all interfaces passive by default.

    ```
    device(config-ipv6-router-ospf-vrf-default-vrf)# default-passive-interface
    ```

Example

The following example sets all OSPFv3 interfaces as passive, causing them to drop all the OSPFv3 control packets.

```
device# configure terminal
device(config)# 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.

> **Note**
> Process restart takes precedence over GR.

Disabling OSPFv3 graceful restart helper

The OSPFv3 graceful restart (GR) helper is enabled by default, and can be disabled on a routing device.

Before You Begin

Procedure

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

   ```
   device# configure terminal
   ```

2. Enter the **ipv6 router ospf** command to enter OSPFv3 router configuration mode and enable OSPFv3 globally.

   ```
   device(config)# ipv6 router ospf
   ```

3. Enter the **no graceful-restart helper** command to disable the GR helper.

   ```
   device(config-ipv6-router-ospf-vrf-default-vrf)# no graceful-restart helper
   ```

Example

The following example disables the GR helper.

```
device# configure terminal
device(config)# ipv6 router ospf
device(config-ipv6-router-ospf-vrf-default-vrf)# no graceful-restart helper
```

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.

Before You Begin

Procedure

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

   ```
   device# configure terminal
   ```

2. Enter the **ipv6 router ospf** command to enter OSPFv3 router configuration mode and enable OSPFv3 globally.

   ```
   device(config)# ipv6 router ospf
   ```

3. 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
   ```

Example

The following example re-enables the GR helper with strict LSA checking.

```
device# configure terminal
device(config)# 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 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 virtual links, so traffic loss is expected during hitless failover.
> - Process restart takes precedence over NSR.

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.

Before You Begin

Procedure

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

   ```
   device# configure terminal
   ```

2. Enter the **ipv6 router ospf** command to enter OSPFv3 router configuration mode and enable OSPFv3 globally.

   ```
   device(config)# ipv6 router ospf
   ```

3. Enter the **graceful restart** command to re-enable GR on the device.

   ```
   device(config-ipv6-router-ospf-vrf-default-vrf)# nonstop-routing
   ```

Example

The following example re-enables NSR for OSPFv3.

```
device# configure terminal
```

```
device(config)# 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.

Before You Begin

> **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.

Procedure

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

   ```
   device# configure terminal
   ```

2. Enter the **ip router-id** command to specify the router ID.

   ```
   device(config)# ip router-id 10.11.12.13
   ```

3. Enter the **ipv6 router ospf** command to enter OSPFv3 router configuration mode and enable OSPFv3 on the device.

   ```
   device(config)# ipv6 router ospf
   ```

4. 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
   ```

5. 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
```

Example

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)# ip router-id 10.11.12.13
device(config)# 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
- 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.

Before You Begin

About This Task

> **Note**
> When IPsec is configured for an area, the security policy is applied to all the interfaces in the area.

Procedure

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

   ```
   device# configure terminal
   ```

2. Enter the **ip router-id** command to specify the router ID.

   ```
   device(config)# ip router-id 10.11.12.13
   ```

3. Enter the **ipv6 router ospf** command to enter OSPFv3 configuration mode and enable OSPFv3 on the device.

   ```
   device(config)# ipv6 router ospf
   ```

4. 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.

Example

The following example enables AH and MD5 authentication for the OSPFv3 area, setting an SPI value of 600.

```
device# configure terminal
device(config)# ip router-id 10.11.12.13
device(config)# 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.

Before You Begin

For IPsec to work, the IPsec configuration must be the same on all the routers to which an interface connects.

About This Task

> **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.

Procedure

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 ethernet 1/1
   ```

3. Enter the **ipv6 ospf area** command to assign a specified area to the interface.

   ```
   device(conf-if-eth-1/1)# ipv6 ospf area 0
   ```

4. Enter **ipv6 ospf authentication spi** *value* **esp null hmac-sha1** and specify a 40-character hexadecimal key.

   ```
   device(conf-if-eth-1/1)# 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.

Example

The following example enables ESP and SHA-1 on a specified OSPFv3 Ethernet interface.

```
device# configure terminal
device(config)# interface ethernet 1/1
device(conf-if-eth-1/1)# ipv6 ospf area 0
device(conf-if-eth-1/1)# 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.

Before You Begin
An OSPFv3 virtual link must be configured.

About This Task

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.

Procedure

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

   ```
   device# configure terminal
   ```

2. Enter the **ip router-id** command to specify the router ID.

   ```
   device(config)# ip router-id 10.1.1.1
   ```

3. Enter the **ipv6 router ospf** command to enter OSPFv3 configuration mode and enable OSPFv3 on the router.

   ```
   device(config)# ipv6 router ospf
   ```

4. Enter **area virtual-link authentication spi** *value* **ah hmac-sha1** *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 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.

Example

The following example configures IPsec on an OSPFv3 area.

```
device# configure terminal
device(config)# ip router-id 10.1.1.1
device(config)# 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 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.

Before You Begin

Procedure

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

   ```
   device# configure terminal
   ```

2. Enter the **ip router-id** command to specify the router ID.

```
device(config)# ip router-id 10.11.12.13
```

3. Enter the **ipv6 router ospf** command to enter OSPFv3 router configuration mode and enable OSPFv3 on the device.

```
device(config)# ipv6 router ospf
```

4. Enter the **key-add-remove-interval** command and specify the desired interval to set the timing of the authentication key add-remove interval.

```
device(config-ipv6-router-ospf-vrf-default-vrf)# key-add-remove-interval 240
```

5. 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
```

Example

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)# ip router-id 10.11.12.13
device(config)# 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.

Before You Begin

About This Task
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.

Procedure

1. Enter the **show ipv6 ospf** command to display general OSPFv3 information.

```
device# show ipv6 ospf
OSPFv3 Process number 0 with Router ID 0xc0a80001(192.168.0.1)
Running 0 days 0 hours 49 minutes 2 seconds
Number of AS scoped LSAs is 1
Sum of AS scoped LSAs Checksum is 0000d72f
External LSA Limit is 250000
Database Overflow Interval is 10
Database Overflow State is NOT OVERFLOWED
Route calculation executed 10 times
Pending outgoing LSA count 0
Authentication key rollover interval 300 seconds
```

```
Number of areas in this router is 3
Router is operating as ABR
Router is operating as ASBR, Redistribute: STATIC
High Priority Message Queue Full count: 0
BFD is disabled, BFD HoldoverInterval: 0
Graceful restart helper is enabled, strict lsa checking is disabled
Nonstop Routing is enabled
Administrative Distance
- External Routes: 110
- Intra Area Routes: 110
- Inter Area Routes: 110
Maximum Paths: 8
```

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.

2. 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       1       1         19      3
```

3. The following example of the **show ipv6 ospf summary all-vrfs total** command shows the cumulative summary output for OSPFv3 sessions.

```
device# show ipv6 ospf summary all-vrfs total
---------------------------------------
        IPv6 OSPF Summary Total
---------------------------------------
Number of instances: 1024
Number of interfaces: 2048
Number of neighbors: 2048
Number of neighbors  in FULL state: 2048
Number of LSAs: 76786
Number of Routes: 67570
```

4. The following example of the **show ipv6 ospf neighbor** command shows OSPFv3 neighbor information for the device.

> **Note**
> QCount is the LSA retransmission list count. The number displays the number of LSAs that are waiting for acknowledgements.

```
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
QCount
192.168.0.2       1 Full      192.168.0.1      192.168.0.2      Eth 0/1        DR        0
```

5. The following example of the **show ipv6 ospf neighbor detail** command shows detailed OSPFv3 neighbor information for the device.

```
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
QCount
192.168.0.2      1 Full    192.168.0.1    192.168.0.2    Eth 0/1      DR      0
                Option: 00-00-00    Timer: 683
BFD State: NONE, BFD HoldoverInterval(sec):Configured: 0 Current: 0
```

6. The following example of the **show ipv6 ospf memory** command shows memory allocation information for OSPFv3.

```
device# show ipv6 ospf memory vrf vrf_1
   Total Dynamic Memory Allocated for this instance : 87046288 bytes
global shared memory pool for all instances
   Memory Type             Size       Allocated  Max-alloc  Alloc-Fails
   MTYPE_OSPF6_AREA        1100       1024       1065       0
   MTYPE_OSPF6_AREA_RANGE  52         0          16         0
   MTYPE_OSPF6_SUMMARY_ADDRE 36       0          16         0
   MTYPE_OSPF6_IF          396        2048       2625       0
   MTYPE_OSPF6_NEIGHBOR    24916      2048       2098       0
   MTYPE_OSPF6_ROUTE_NODE  36         71666      72415      0
   MTYPE_OSPF6_ROUTE_INFO  52         71666      80537      0
   MTYPE_OSPF6_PREFIX      24         0          16         0
   MTYPE_OSPF6_LSA         252        76787      133135     0
   MTYPE_OSPF6_VERTEX      196        5120       5327       0
   MTYPE_OSPF6_SPFTREE     60         1024       1056       0
   MTYPE_OSPF6_NEXTHOP     32         5134       8192       0
   MTYPE_OSPF6_EXTERNAL_INFO 52       0          1024       0
   MTYPE_THREAD            68         14703      15192      0
   MTYPE_OSPF6_LINK_LIST   44         6849444    7050698    0
   MTYPE_OSPF6_LINK_NODE   28         170996     265654     0
   MTYPE_OSPF6_LSA_RETRANSMI 20       0          25598      0
Global Memory Pool Usage for all instances : 415468328 bytes
global Heap memory for all instances
   Memory Type             Size       Allocated  Max-alloc  Alloc-Fails
   MTYPE_OSPF6_TOP         41104      1024       1024       0
   MTYPE_OSPF6_LSA_HDR     416        76787      107723     0
   MTYPE_OSPF6_RMAP_COMPILED 0        0          0          0
   MTYPE_OSPF6_OTHER       96         132591     137421     0
   MTYPE_THREAD_MASTER     200        1024       1024       0
--------------------
Packet Tx thread Info
--------------------
Queue Id[0]: Enqueued[291763] Dequeued [291763]
Queue Id[1]: Enqueued[13108] Dequeued [13108]
Send Failed Packets - 0
device#
```

7. 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: Eth 0/1
  Inactive interface(s)attached to this area: None
```

```
   Number of Area scoped LSAs is 6
   Sum of Area LSAs Checksum is 000370d3
   Statistics of Area 0:
     SPF algorithm executed 7 times
     SPF last updated: 145 sec ago
     Current SPF node count: 3
       Router: 2 Network: 1
       Maximum of Hop count to nodes: 2
```

8. 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)
Eth 0/1        0                up     BCST 1     DR        1/1
Lo 1           1                up     BCST 1     Loopback 0/0
Lo 2           2                up     BCST 1     Loopback 0/0
```

9. 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     192.168.0.2     2018:6::5                       Full       Eth 0/1
              Option: 00-00-00    QCount: 0    Timer: 578
```

10. 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                192.168.0.2      2018:6::4                     P2P
  Timer intervals(sec) :
    Hello 10, Hello Jitter 10,  Dead 40, Retransmit 5, TransmitDelay 1
  DelayedLSAck:     3 times
  Authentication: Not Configured
  Statistics:
      Type    tx         rx         tx-byte    rx-byte
      Unknown 0          0          0          0
      Hello   9          9          356        360
      DbDesc  3          2          104        136
      LSReq   1          1          64         28
      LSUpdate 8         7          600        784
      LSAck   7          5          312        240
      OSPF messages dropped,no authentication: 0
  Neighbor: State: Full Address: 2018:6::5 Interface: Eth 0/1
```

11. 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
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
2              Typ7 1           192.168.0.1      80000002 211  f5fd  44    Yes
     Bits: EF-
```

```
             Metric: 10
             Prefix Options:
             Referenced LSType: 0
             Prefix: ::/0
             Forwarding-Address: 4321::56
2                Typ7 2            192.168.0.1    80000001 143  a5b2  60     Yes
             Bits: EF-
             Metric: 1
             Prefix Options:
             Referenced LSType: 0
             Prefix: 9876::5/128
             Forwarding-Address: 4321::56
```

12. The following example of the **show ipv6 ospf routes** command shows output for OSPFv3 routes.

```
device# show ipv6 ospf routes
  Current Route count: 3
    Intra: 3 Inter: 0 External: 0 (Type1 0/Type2 0)
    Equal-cost multi-path: 0
    OSPF Type: IA- Intra, OA - Inter, E1 - External Type1, E2 - External Type2
  Destination                     Cost      E2Cost     Tag         Flags    Dis
IA 1234::56/128                   0         0          0           00000002 110
    Next_Hop_Router               Outgoing_Interface Adv_Router
    ::                            Lo 1               192.168.0.1
  Destination                     Cost      E2Cost     Tag         Flags    Dis
IA 2018:6::/64                    1         0          0           00000003 110
    Next_Hop_Router               Outgoing_Interface Adv_Router
    ::                            Eth 0/1            192.168.0.1
  Destination                     Cost      E2Cost     Tag         Flags    Dis
IA 4321::56/128                   0         0          0           00000002 110
    Next_Hop_Router               Outgoing_Interface Adv_Router
    ::                            Lo 2               192.168.0.1
```

13. The following example of the **show ipv6 ospf database as-external** command shows information about external LSAs.

```
device# show ipv6 ospf database as-external
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
N/A            Extn 2         192.168.0.1      80000001 221  d72f  44     Yes
     Bits: E--
     Metric: 1
     Prefix Options:
     Referenced LSType: 0
     Prefix: 9876::5/128
```

14. 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 2         192.168.0.1      80000001 790  8399  56    Yes
0               Link 2         192.168.0.2      80000001 652  e939  56    Yes
0               Rtr  0         192.168.0.1      80000003 309  2b16  40    Yes
0               Rtr  0         192.168.0.2      80000002 649  1e26  40    Yes
0               Net  2         192.168.0.1      80000001 648  eafb  32    Yes
0               Inap 1         192.168.0.1      80000001 790  2d79  44    Yes
0               Inap 10        192.168.0.1      80000001 309  0b74  44    Yes
0               Iap  60        192.168.0.1      80000001 648  96dd  44    Yes
1               Rtr  0         192.168.0.1      80000002 309  23a0  24    Yes
1               Inap 4         192.168.0.1      80000001 790  1c2d  36    Yes
1               Inap 11        192.168.0.1      80000001 309  017d  44    Yes
1               Iap  0         192.168.0.1      80000002 1233 e227  52    Yes
2               Rtr  0         192.168.0.1      80000001 309  4975  24    Yes
2               Inap 7         192.168.0.1      80000001 309  f0af  44    Yes
2               Inap 9         192.168.0.1      80000001 309  e95a  36    Yes
2               Typ7 1         192.168.0.1      80000002 309  f5fd  44    Yes
2               Typ7 2         192.168.0.1      80000001 241  a5b2  60    Yes
2               Iap  0         192.168.0.1      80000001 309  0ede  52    Yes
N/A             Extn 2         192.168.0.1      80000001 241  d72f  44    Yes
```

15. The following example of the **show ipv6 ospf spf tree** command with the **tree** shows information about the SPF trees.

```
device# show ipv6 ospf spf tree
SPF tree for Area 0
+- 192.168.0.1 cost 0
    +- 192.168.0.1:2(N) cost 1
        +- 192.168.0.2:0 cost 1

SPF tree for Area 1
+- 192.168.0.1 cost 0

SPF tree for Area 2
+- 192.168.0.1 cost 0
```

16. The following example of the **show ipv6 ospf spf table** command with the **table** shows information about the SPF table.

```
device# show ipv6 ospf spf table
SPF table for Area 0
  Destination          Bits Options  Cost  Nexthop                    Interface
R 192.168.0.2          ----- V6E---R--   1  fe80::629c:9fff:fe5a:8919 Eth 0/1
N 192.168.0.1[2]       ----- V6E---R--   1  ::                         Eth 0/1

SPF table for Area 1
  Destination          Bits Options  Cost  Nexthop                    Interface

SPF table for Area 2
  Destination          Bits Options  Cost  Nexthop                    Interface
```

17. 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
2     9876::5/128                               Static    Type-2      1
```

18. The following example of the **show ipv6 ospf routes** command shows information about a
specified OSPFv3 route.

```
device# show ipv6 ospf routes 1234::56/128
   Destination                      Cost       E2Cost      Tag         Flags    Dis
IA 1234::56/128                     0          0           0           00000002 110
   Next_Hop_Router                  Outgoing_Interface Adv_Router
   ::                               Lo 1               192.168.0.1
```

Supported scale for OSPFv3 and performance considerations

Scalability of OSPFv3 gets limited by availability of resources like Memory, CPU and hardware tables.
While increasing the scale, you need to consider time taken for adjacency formation and route
convergence.

The supported scale for OSPFv3 across all VRF instances is as follows:

Table 40: Supported scale across all VRFs

Description	Number supported
Maximum number of OSPFv3 VRF Instances	1024
Maximum number of OSPFv3 Areas	1024
Number of OSPFv3 Interfaces	2048
Number of OSPFv3 Neighbors	2048
Maximum Number of OSPFv3 Routes	64k (65536)

The supported scale for OSPFv3 per VRF instance is as follows:

Table 41: Supported scale per VRF

Description	Number supported
Maximum number of OSPFv3 Areas	10
Number of OSPFv3 Interfaces	256
Number of OSPFv3 Neighbors	256
Maximum Number of OSPFv3 Routes	64k (65536)

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.

Some vendors also use the terms Multi-VRF CE or VRF-Lite for this technology. VRF-Lite provides a reliable mechanism for a network administrator to maintain multiple virtual routers on the same device. The goal of providing isolation among different VPN instances is accomplished without the overhead of heavyweight protocols (such as MPLS) used in secure VPN technologies. Overlapping address spaces can be maintained among the different VPN 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.

········	Sales	··················	Operations
------	Marketing	·· ·· ·· ··	Partner A
·—·—·—	Finance	—·—··—	Partner B
------	Engineering	————	Combined Network Traffic
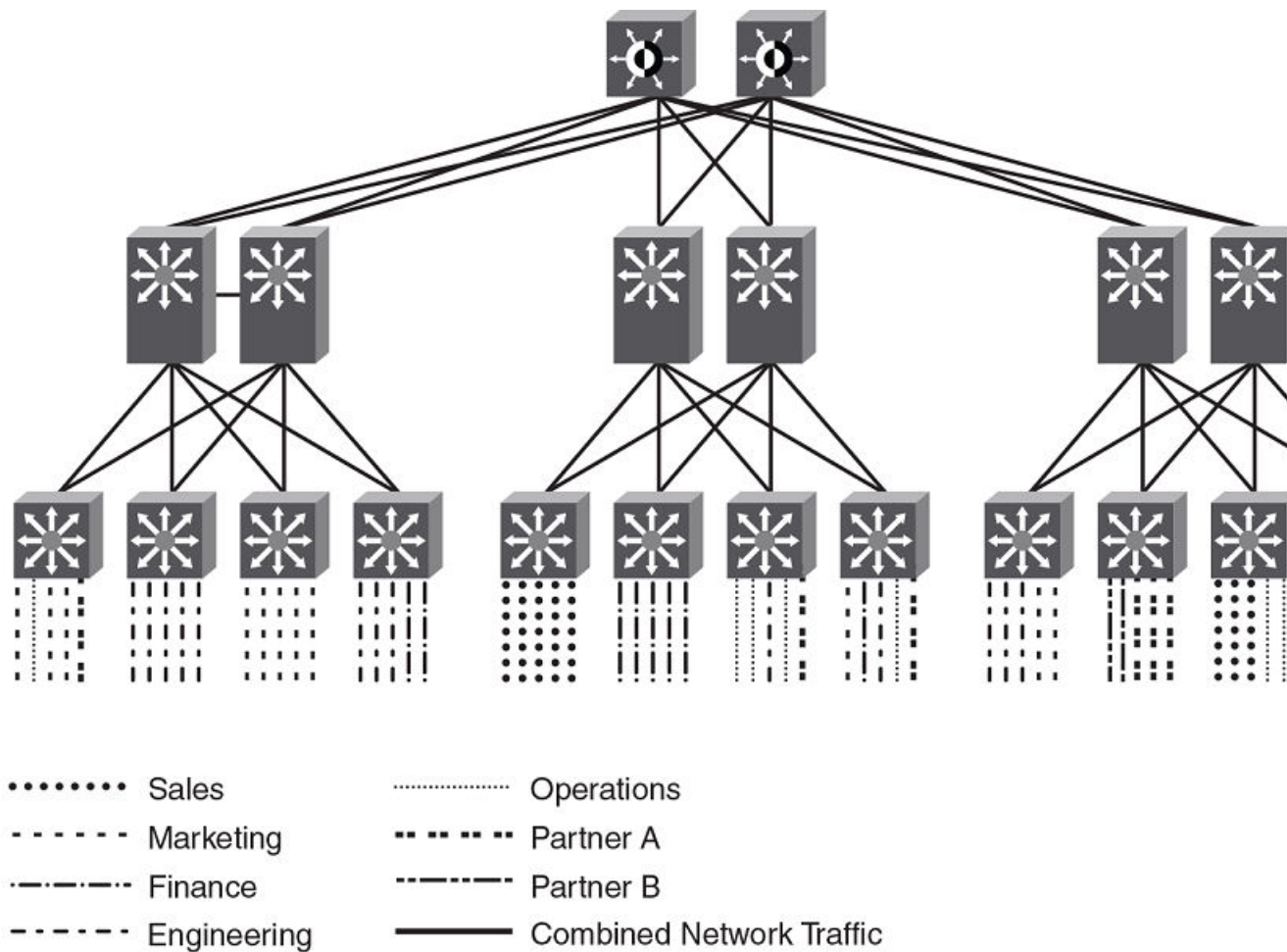
Figure 55: Example high-level Multi-VRF topology

A Multi-VRF instance can be configured on any of the following:

- Virtual interfaces
- Loopback interfaces
- Ethernet interfaces

To configure Multi-VRF, perform the following steps:

- Configure 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.

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 this chapter.

Configuring Multi-VRF

Configuring a VRF instance

Do the following to configure a VRF instance.

About This Task

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. 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, and static route, IPv6 neighbor, and multicast for IPv6.

> **Important**
> 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.

Procedure

1. Enter global configuration mode and create a VRF instance.

   ```
   device# configure terminal
   device(config)# vrf corporate
   device(config-vrf-corporate)#
   ```

2. (Optional) Assign a router ID.

   ```
   device(config-vrf-corporate)# ip router-id 1.1.1.1
   ```

3. 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
   ```

4. Verify the configuration.

   ```
   device(config-vrf-corporate)# do show vrf
   Total number of VRFs configured: 4
   VrfName                     VrfId  V4-Ucast  V6-Ucast
   corporate                   3      Enabled   Disabled
   default-vrf                 1      Enabled   Enabled
   mgmt-vrf                    0      Enabled   Enabled
   test1                       2      Enabled   Enabled
   ```

Starting a routing process for a VRF

You must enable a routing protocol for each VRF instance. This example uses OSPF.

Procedure

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.

About This Task

➡ **Important**
After a VRF instance is configured on the device, one or more Layer 3 interfaces (physical or virtual Ethernet) must be assigned to the VRF. This causes all existing IP addresses to be deleted; this action also triggers cache deletion, route deletion, and associated cleanup. After assigning an interface to the VRF, the user must reconfigure the IP address and interface properties.

Procedure

1. Enter global configuration mode.

```
device(config)# configure terminal
```

2. Enter the **interface ve** command to specify a virtual Ethernet (VE) interface and enter VE configuration mode.

```
device(config)# interface ve 10
```

3. In VE configuration mode, enable forwarding for the VRF "guest".

```
device(config-if-Ve-10)# vrf forwarding guest
```

4. Configure an IPv4 address and mask on the VE interface.

```
device(config-if-Ve-10)# ip address 192.168.1.254/24
```

5. Enable OSPF Area 0.

```
device(config-if-Ve-10)# ip ospf area 0
```

6. Configure the interface as passive.

```
device(config-if-Ve-10)# ip ospf passive
```

7. Exit the configuration.

```
device(config-if-Ve-10)# exit
```

Assigning a loopback interface to a VRF

Do the following to assign a loopback interface to a nondefault VRF.

About This Task
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.

Procedure

1. Enter global configuration mode.

```
device# configure terminal
```

2. Enter the **interface loopback** command to specify a loopback interface and enter interface loopback configuration mode.

```
device(config)# interface loopback 1
device(config-Loopback-1)#
```

3. Use the **vrf forwarding** command to assign the interface to the VRF "customer-1" in this example.

```
device(config-Loopback-1)# vrf forwarding customer-1
```

4. Assign an IPv4 address and mask to the loopback interface.

```
device(config-Loopback-1)# ip address 10.0.0.1/32
```

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.

Example

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: 4
VrfName                         VrfId  V4-Ucast   V6-Ucast
corporate                       3      Enabled    Disabled
default-vrf                     1      Enabled    Enabled
mgmt-vrf                        0      Enabled    Enabled
test1                           2      Enabled    Enabled
```

Example

To verify a specific VRF, enter the **show vrf** *vrf-name* command, as in the following example.

```
device# show vrf corporate
VRF-Name: corporate, VRF-Id: 3
IP Router-Id: 1.1.1.1
Interfaces:
       Lo 10
Address-family IPV4 unicast
       Max routes:-   Route count:0
   No import route-maps
   No export route-maps

Address-family IPV6 unicast
       Max routes:-   Route count:2
   No import route-maps
```

```
        No Export route-maps
```

Use the **show vrf detail** command to display information about all VRFs.

Example

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 42: Additional useful show commands

Default VRF	User-defined VRF
show ip route	**show arp vrf**
show ip ospf neighbor	**show ip route vrf** *vrf-name*
show ip bgp summary	**show ip ospf neighbor vrf** *vrf-name*
	show ip bgp summary vrf *vrf-name*

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.

Example

To delete a VRF instance from a specific port, use the **no** form of the **vrf forwarding** 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(conf-if-eth-0/1)# no vrf forwarding
```

Example
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
```

Example

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
```

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.

About This Task

If this command is not used, the maximum number of routes is 4294967295. This number does not appear in a running configuration.

Procedure

1. Enter global configuration mode.

   ```
   device# configure terminal
   ```

2. Specify a VRF instance (in this example, "myvrf") and enter VRF configuration mode.

   ```
   device(config)# vrf myvrf
   device(config-vrf-myvrf)#
   ```

3. 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
   ```

4. 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
   ```

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 or OSPF, 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 56: eBGP configured between PE1 and PE2 with OSPF (Area 0) configured between PEs and CEs
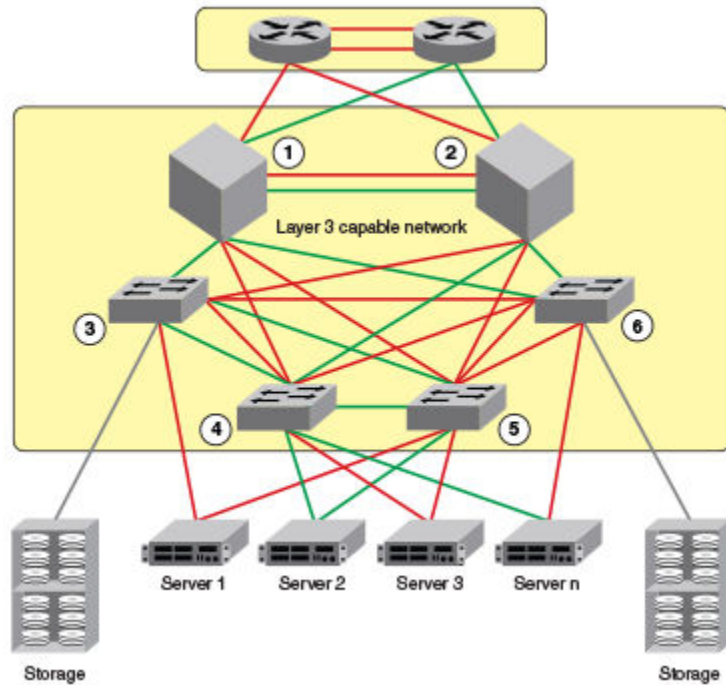
1. PE1
2. PE2
3. CE1
4. CE2
5. CE3
6. CE4

Topology details are listed below.

Table 43: Topology details

Node	Description	Networks	Carries routes . . .	Interfaces
PE1	Aggregation	10.1.1.0/24 10.3.1.0/24		0/1 0/2 0/3
PE2	Aggregation	10.2.1.0/24 10.3.1.0/24		0/1 0/2 0/3
CE1	Edge	10.1.1.0/24	10.1.2.0/24 10.1.3.0/24	0/1 0/2
CE2	Edge	10.1.1.0/24	10.1.2.0/24 10.1.3.0/24	0/1 0/2

Table 43: Topology details (continued)

Node	Description	Networks	Carries routes . . .	Interfaces
CE3	Edge	10.2.1.0/24	10.2.2.0/24 10.2.3.0/24	0/1 0/2
CE4	Edge	10.2.1.0/24	10.2.2.0/24 10.2.3.0/24	0/1 0/2

- 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.
- 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").

Multi-VRF with eBGP and OSPF: Configuring PE1

Two VRFs ("red" and "green") are defined. In the eBGP configuration, PE1 is defined in Local AS 1.

About This Task

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").

Procedure

1. In global configuration mode, create VLANs 10 and 20.

   ```
   device(config)# vlan 10
   device(config-Vlan-10)# exit
   device(config)# vlan 20
   ```

2. From global configuration mode, enter interface subtype configuration mode, and then create a virtual Ethernet routing interface for the VLAN.

```
device(config)# vlan 10
device(config-Vlan-10)# router-interface ve 10
```

3. Repeat the above steps as appropriate for remaining physical, VLAN, and virtual Ethernet interfaces.
4. Create VRF "green".

```
device(config)# vrf green
device(config-vrf-green)# exit
```

Repeat for every VRF instance. 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)# vrf red
device(config-vrf-red)# exit
```

6. In global configuration mode, enable BGP routing and configure the following in this IPv4 example.
   a. Enable BGP routing.

```
device(config)# 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 for VRF "red."

```
device(config)# 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)# 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)# 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)# interface ethernet 0/1
    device(conf-if-eth-0/1)# vrf forwarding green
    ```

    b. Assign Area 0.

    ```
    device(conf-if-eth-0/1)# ip ospf area 0
    ```

    c. Assign an IPv4 network.

    ```
    device(conf-if-eth-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(conf-if-eth-0/2)# interface ethernet 0/2
    device(conf-if-eth-0/2)# vrf forwarding red
    device(conf-if-eth-0/2)# ip ospf area 0
    device(conf-if-eth-0/2)# ip address 10.1.1.1/24
    device(conf-if-eth-0/2)# exit
    ```

11. Configure the VE interfaces for the appropriate VRF and network.

    a. Configure VE 10, corresponding to VLAN 10.

    ```
    device(config)# interface ve 10
    device(config-if-Ve-10)# vrf forwarding green
    device(config-if-Ve-10)# ip address 10.3.1.1/24
    device(config-if-Ve-10)# no shutdown
    ```

    b. Repeat the above for VE 20, corresponding to VLAN 20.

    ```
    device(config-ve-10)# interface ve 20
    device(config-if-Ve-20)# vrf forwarding red
    device(config-if-Ve-20)# ip address 10.3.1.1/24
    device(config-if-Ve-20)# no shutdown
    device(config-if-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.

Example

The following summarizes the configuration on PE2.

```
device(config)# vlan 10
device(config-vlan-10)# router-interface ve 10
device(config-vlan-10)# exit
device(config)# vlan 20
device(config-vlan-20)# router-interface ve 20
device(config-vlan-20)# exit
device(config)# vrf green
device(config-vrf-green)# exit
device(config)# vrf red
device(config-vrf-red)# exit
device(config)# 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 10.2.1.1 remote-as 2
device(config-bgp-ipv4u-vrf)# network 10.2.1.0/24
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)# 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)# 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)# interface ethernet 0/2
device(conf-if-eth-0/2)# vrf forwarding green
device(conf-if-eth-0/2)# ip ospf area 0
device(conf-if-eth-0/2)# ip address 10.2.1.1/24
device(conf-if-eth-0/2)# interface ethernet 0/3
device(conf-if-eth-0/3)# vrf forwarding red
device(conf-if-eth-0/3)# ip ospf area 0
device(conf-if-eth-0/3)# ip address 10.3.1.1/24
device(conf-if-eth-0/3)# exit
device(config)# interface ve 10
device(config-Ve-10)# vrf forwarding green
device(config-Ve-10)# ip address 10.2.1.1/24
device(config-Ve-10)# exit
device(config)# 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 loopback interfaces, allowing them to carry routes from these networks.

Procedure

1. Enable OSPF routing.

```
device(config)# 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.

```
device(config)# interface loopback 1
device(config-Loopback-1)# ip address 10.1.1.1/32
```

5. Configure an Ethernet interface, assign it to Area 0, and assign it to the appropriate network.

```
device(config-Loopback-1)# interface ethernet 0/1
device(conf-if-eth-0/1)# ip ospf area 0
device(conf-if-eth-0/1)# ip address 10.1.2.1/24
```

6. Repeat the above for additional loopback and Ethernet interfaces.

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.

Example

The following summarizes the configuration.

```
device(config)# 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)# interface loopback 1
device(config-Loopback-1)# ip address 10.1.1.1/32
device(config-Loopback-1)# exit
device(config)# interface loopback 2
device(config-Loopback-2)# ip address 10.1.1.2/32
device(config-Loopback-2)# exit
device(config)# interface ethernet 0/1
device(conf-if-eth-0/1)# ip ospf area 0
device(conf-if-eth-0/1)# ip address 10.2.2.1/24
device(config)# interface ethernet 0/2
device(conf-if-eth-0/2)# ip ospf area 0
device(conf-if-eth-0/2)# ip address 10.2.3.1/24
```

Inter-VRF Route Leaking

VRFs operate without knowledge of one another unless they are imported or exported into one another by means of 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.

Inter-VRF route leaking is useful in cases where multiple VRFs share the same path to an external domain, while maintaining the internal routing information limited to their own VRFs. This feature enables a data center to consolidate multiple VRF services onto one server.

Both static and dynamic route leaking are supported. Each routed interface (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. You can leak routes by configuring a route map and associating this route map with a source VRF. The match criteria defined in the route map consist of specific route prefixes that exist in the source VRF.

> **Important**
> When OptiScale™ is enabled for the SLX 9540, the non-default VRF is not supported, nor is inter-VRF route leaking. Multi-VRF and inter-VRF route leaking are supported only when OptiScale is not enabled. For more information, see Internet Route Scaling on page 475.

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.
- Routes in the management-vrf with a next-hop as eth0 or a management interface are not leaked.

Inter-VRF route conflicts

> **Important**
> Deploy this feature only if you are an advanced user, because route leak configuration in source VRFs can collide with route and interface definitions in target VRFs. Such a situation can lead to unpredictable behavior in packet forwarding.

Leaked route conflicts can occur in situations such as the following:

- Static route conflict
- Dynamic route conflict
- Connected route conflict

A static route conflict can occur when the same prefix is reachable by two different next hops in the target VRF. The forwarding behavior is different depending on which command occurred later. In the following example, the global configuration presents a static route conflict for 10.1.2.0/24.

```
vrf red
device(config-vrf-red)# address-family ipv4 unicast
```

```
device(config-vrf-red-ipv4-unicast)# ip route 10.1.2.0/24 next-hop-vrf green 10.1.1.1
device(config)# vrf green
device(config-vrf-green)# address-family ipv4 unicast
device(config-vrf-green)# 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 (where the most recent configuration takes precedence) does not apply. The order of preference is as follows: (1) connected, (2) static, (3) OSPF, and (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, or 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.

> **Important**
>
> 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.

The following example illustrates a connected route conflict:

```
device(config)# vrf red
device(config-vrf-red)# address-family ipv4 unicast
device(vrf-red-ipv4-unicast)# ip route 10.1.2.0/24 next-hop-vrf green 10.1.1.1
device(config)# interface ethernet 0/1
device(conf-if-eth-0/1)# vrf forwarding green
device(conf-if-eth-0/1)# ip address 10.1.2.1/24
```

> **Note**
>
> Be aware of such possible conflicts before deploying the route leak feature, because there is no method for error checking these scenarios. A best practice is to ensure that definitions are globally unique and route collisions do not exist.

Displaying inter-VRF route leaking

The **show ip route** command displays "%vrf" in the route type, with the next-hop address, for the leaked routes in a VRF.

The following example displays output for routes in the default-vrf

```
device# show ip route
IP Routing Table for VRF "default-vrf"
Total number of IP routes: 6
'*' denotes best ucast next-hop
'[x/y]' denotes [preference/metric]

1.1.1.0/24,
    *via 8.8.8.100%vrf200, Ve200 , [1/1], 2m45s, static, tag 0
2.2.2.0/24, attached
    *via DIRECT, Ve 100, [0/0], 2m45s, direct, tag 0
2.2.2.100/32, attached
    *via DIRECT, Ve 100, [0/0], 2m45s, local, tag 0
40.0.0.0/24, attached
    *via DIRECT, Eth 0/2, [0/0], 31m15s, direct, tag 0
```

```
40.0.0.1/32, attached
    *via DIRECT, Eth 0/2, [0/0], 31m15s, local, tag 0
90.0.0.0/24,
    *via 40.0.0.100, Eth 0/2, [1/1], 16m42s, static, tag 0
```

You can also determine the leaked route for a specific VRF by using the **show ip route vrf** command, as illustrated in the following example for vrf1 that displays "%default-vrf" in the route type.

```
device# show ip route vrf vrf1
IP Routing Table for VRF "vrf1"
Total number of IP routes: 6
'*' denotes best ucast next-hop
'[x/y]' denotes [preference/metric]

3.3.3.0/24,
    *via 14.14.14.100, Ve 2, [1/1], 0m23s, static, tag 0
11.11.11.0/24,
    *via 40.0.0.2%default-vrf, Eth 0/2, [1/1], 1m40s, static, tag 0
14.14.14.0/24, attached
    *via DIRECT, Ve 2, [0/0], 9m11s, direct, tag 0
14.14.14.14/32, attached
    *via DIRECT, Ve 2, [0/0], 9m11s, local, tag 0
15.15.15.0/24, attached
    *via DIRECT, Ve 3, [0/0], 9m11s, direct, tag 0
15.15.15.1/32, attached
    *via DIRECT, Ve 3, [0/0], 9m11s, local, tag 0
```

Configuring static inter-VRF route leaking

Use the following procedure to configure static inter-VRF route leaking.

About This Task

➡ **Important**
Static inter-VRF route leaking is a feature that should be deployed only by an advanced user.

Procedure

1. Create the VRF instances you want to be the leaker (source VRF) and where the route is being leaked to (destination VRF).
2. Specify the interface for the source VRF and map it to the source VRF.
3. Enter the IP address/mask to be used for this VRF instance.
4. Specify the interface you want to be the destination VRF and map it to the destination VRF.
5. Specify the IP address/mask to receive the leaked route.
6. Change the configuration mode to source VRF address-family context.
7. Configure the route to be leaked, specifying the route prefix, the next-hop VRF name as the destination VRF and the next hop to the destination VRF.
8. (Optional) (Optional) For bidirectional inter-VRF route leaking, repeat the above steps, but swap the source and destination addresses.

Example of static inter-VRF route leaking

In this example, one of the static routes in VRF "Blue" (10.50.2.0/24) is being allowed to communicate with one in VRF "Green" (10.55.2.0/24).

About This Task

The ovals represent virtual partitions (VRFs) in the router. The destination VRF ("Green") is where the route is being leaked to, and the source VRF ("Blue") is where the route is being leaked from.
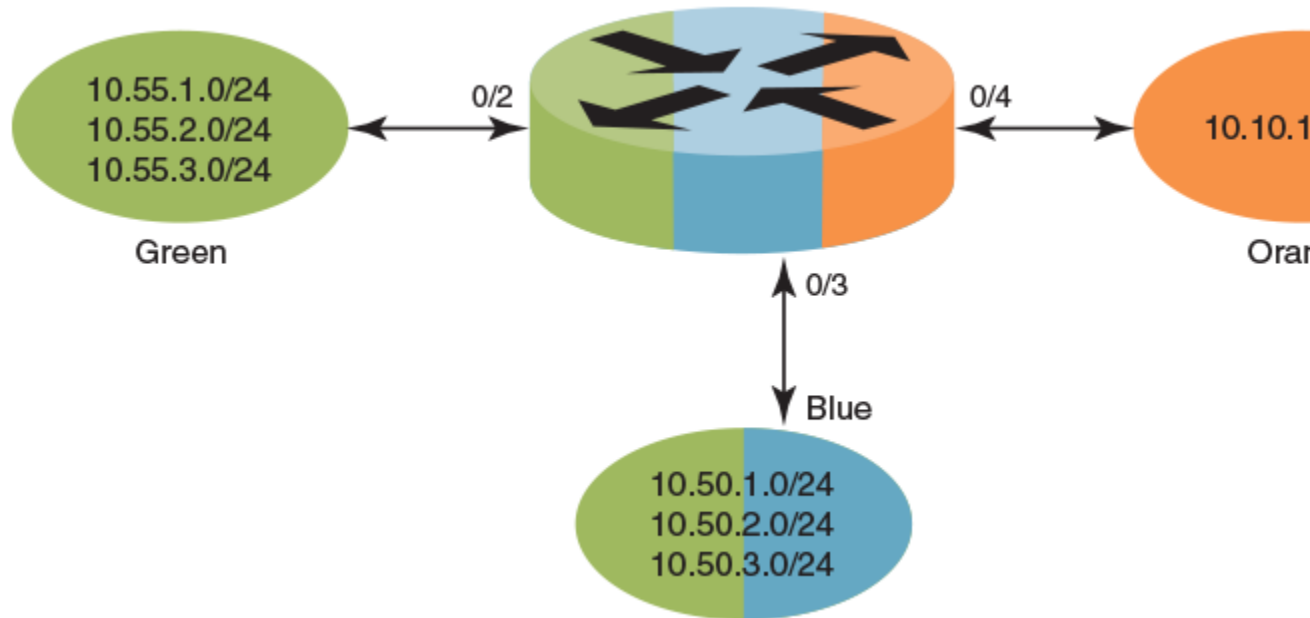
Figure 57: Static inter-VRF route leaking

Procedure

1. In global configuration mode, create and configure VRF "Green".

   ```
   device(config)# vrf Green
   device(conf-vrf-Green)# address-family ipv4 unicast
   device(vrf-Green-ipv4-unicast)#
   ```

2. Repeat the above for VRF "Blue".

   ```
   device(config)# vrf Blue
   device(conf-vrf-Blue)# address-family ipv4 unicast
   device(vrf-Blue-ipv4-unicast)#
   ```

3. Configure an interface in the destination VRF "Green" by using the using the **vrf forwarding** command and configuring a corresponding IP address and subnet mask.

   ```
   device(config)# interface eth 0/2
   device(conf-eth-0/2)# vrf forwarding Green
   device(conf-eth-0/2)# ip address 10.55.1.2/24
   ```

4. Repeat the above for the source VRF "Blue", with an appropriate interface and network.

   ```
   device(config)# interface eth 0/3
   device(conf-eth-0/3)# vrf forwarding Blue
   device(conf-eth-0/3)# ip address 10.50.1.2/24
   ```

5. Enter address-family IPv4 unicast configuration mode for the source VRF address family context for configuring static route leak.

```
device(config)# vrf Blue
device(conf-vrf-Blue)# 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 instance 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 *Extreme SLX-OS Command Reference* for details on the **ip route next-hop-vrf** command.

```
device(vrf-Blue-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 instance and the default-vrf in the destination VRF.

```
device(vrf-Blue-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 "Blue".

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 not be 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).

About This Task

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 the "DHCPv4" chapter.)
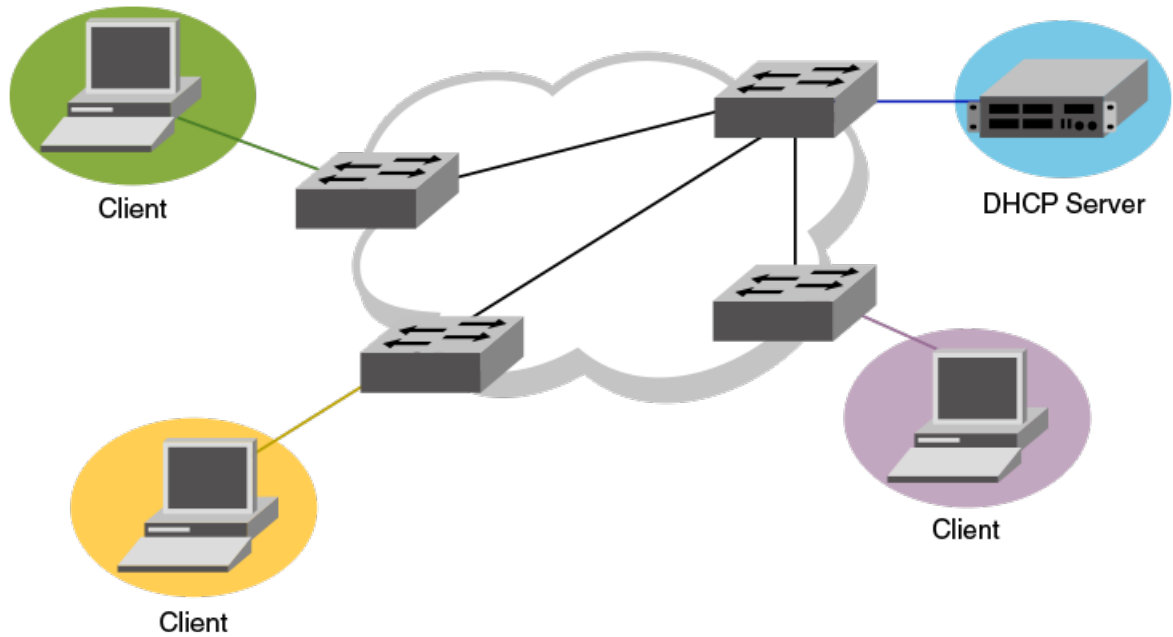
Figure 58: Inter-VRF route leaking example for connecting clients to a DHCP server in a different VRF.

The following example shows setting up Inter-VRF route leaking and DHCP between the red VRF and the blue VRF.

> **Note**
>
> Inter-VRF route leaking supports both IPv4 and IPv6. Use the **ip address** and **ip route** commands for IPv4 and the **ipv6 address** and **ipv6 route** commands for IPv6. These commands support IP addresses, Ethernet interfaces, and virtual Ethernet (VE) interfaces for the leak destination. Refer to the *Extreme SLX-OS Command Reference*.

Procedure

1. Configure VRF forwarding on a VE interface.

```
device(config)# interface ve 100
device(config-if-Ve-100)# no shutdown
device(config-if-Ve-100)# vrf forwarding red
      <- interface is in VRF "red" ->
device(config-if-Ve-100)# ip address 10.1.1.1/24
device(config-if--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)# vrf red
device(conf-vrf-red)# address-family ipv4 unicast
device(vrf-red-ipv4-unicast)# 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 basic procedure to configure dynamic Inter-VRF route leaking.

About This Task

➡️ **Important**
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.

Procedure

1. Configure the VRF instances you want to be the leaker (source VRF) and where the route is being leaked to (destination VRF).
2. Specify the interface for the source VRF and map it to the source VRF.
3. Enter the IP address/mask to use for this VRF instance.
4. Specify the interface you want to be the destination VRF and map it to the destination VRF.
5. Specify the IP address/mask to receive the leak.
6. Configure the route map and associated prefix-list.
7. Change the configuration mode to destination VRF address family context. (IPv4 and IPv6 are supported.)
8. Configure the import command, specifying the source VRF and route map to be leaked.
9. (Optional) You can leak BGP or OSPF routes that were learned by the source VRF into the destination VRF. Static routes can also be leaked.

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.

About This Task

The figure below depicts a typical dynamic route-leaking scenario. VRFs "Yellow" and "Green" are virtual partitions in the same router. The destination VRF ("Green") is where the route is being leaked to, and the source VRF ("Yellow") is where the route is being leaked from. In this example, IPv4 is used.
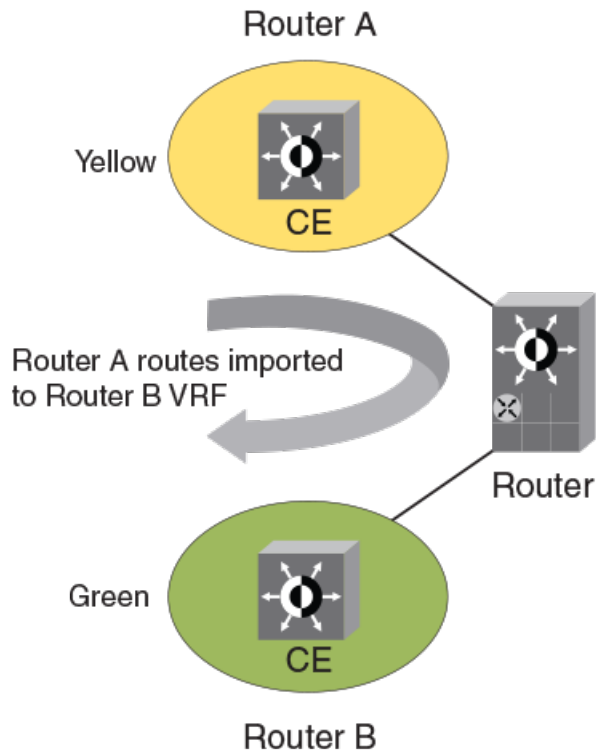
Figure 59: Dynamic inter-VRF route leaking

Procedure

1. Configure VRF "Green".

```
device(config)# vrf Green
device(config-vrf-Green)# address-family ipv4 unicast
```

2. Configure VRF "Yellow".

```
device(config)# vrf Yellow
device(config-vrf-Yellow)# address-family ipv4 unicast
```

3. Configure an IPv4 prefix list, named "import-prefix" in this example.

```
device(config)# ip prefix-list import-prefix permit 10.2.3.0/24
device(config)# ip prefix-list import-prefix permit 10.1.2.0/24
```

4. Configure a route map with "match" conditions, including metric and tag for matched routes, and then import it.

```
device(config)# route-map import-map permit 10
device(config-route-map-import-map/permit/10)# match metric 10
device(config-route-map-import-map/permit/10)# match tag 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)# 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) Redistribute any routes learned by OSPF (or BGP) in the source VRF into the destination VRF. The following shows an OSPF example.

```
device(config-ipv4-unicast)# exit
device(config-vrf-Green)# exit
device(config)# 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 leaking are included in the following table and described in detail in the *Extreme SLX-OS Command Reference*.

Table 44: 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.	Global 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.	Global 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.	OSPF VRF router configuration
redistribute bgp	Redistributes leaked BGP routes that were imported to another VRF into BGP of this VRF instance.	• Address-family ipv4 unicast • Address-family ipv6 unicast • Address-family ipv4 unicast vrf • Address-family ipv6 unicast vrf

Note
The **redistribute** commands 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.

Internet Route Scaling

Internet route scaling (OptiScale™ for Internet Routing) helps to ensure reachability for all networks in the internet routing table and eliminates unnecessary table entries.

OptiScale creates an enhanced route scale profile using the following methods:

- FIB (Forwarding Information Base) optimization: Protocol routes are downloaded to the RIB (Routing Information Base), which then downloads them to the FIB. Initial route optimization eliminates redundant next-hop entries.
- Hardware table optimization: Routes with common prefixes are stored in separate tables. The Longest Exact Match (LEM) table handles common prefixes. The Longest Prefix Match (LPM) table retains enough headroom to accommodate route entries that are added.

OptiScale is supported as follows:

- It cannot coexist with uRPF (unicast Reverse Path Forwarding).
- (SLX 9640) It is supported in the default and non-default VRF.
- (SLX 9540) It is supported only in the default VRF. The SLX 9540 supports the default VRF only when OptiScale is enabled.

For more information, see the *OptiScale™ for Internet Routing* white paper at https://www.extremenetworks.com/resources/white-paper/optiscale-for-internet-routing-internet-scale-routing-on-slx-border-leaf-and-border-platforms/.

Configure the Profile Route

Internet route scaling (OptiScale™ for Internet Routing) creates an enhanced route scale profile to help ensure reachability for all networks in the internet routing table and to eliminate unnecessary table entries.

Procedure

1. Access global configuration mode.

```
device# configure terminal
```

2. Access configuration hardware mode.

```
device(config)# hardware
```

3. Enable the optiscale feature.

```
device(config-hardware)# profile route multi_vrf
```

4. Enable FIB compression for IPv4 or IPv6.

```
device(config-hardware)# profile route route-enhance v4_fib_comp
device(config-hardware)# profile route route-enhance v6_fib_comp
```

5. Enable route enhancement for hardware optimization.

```
device(config-hardware)# profile route route-enhance hw_opt
```

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.
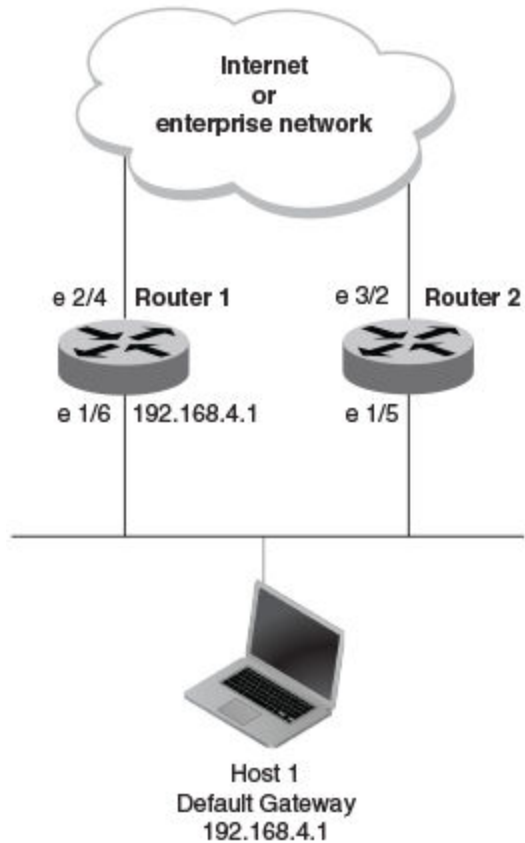
Figure 60: Single point of failure with Device 1 being the Host1 default gateway

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.
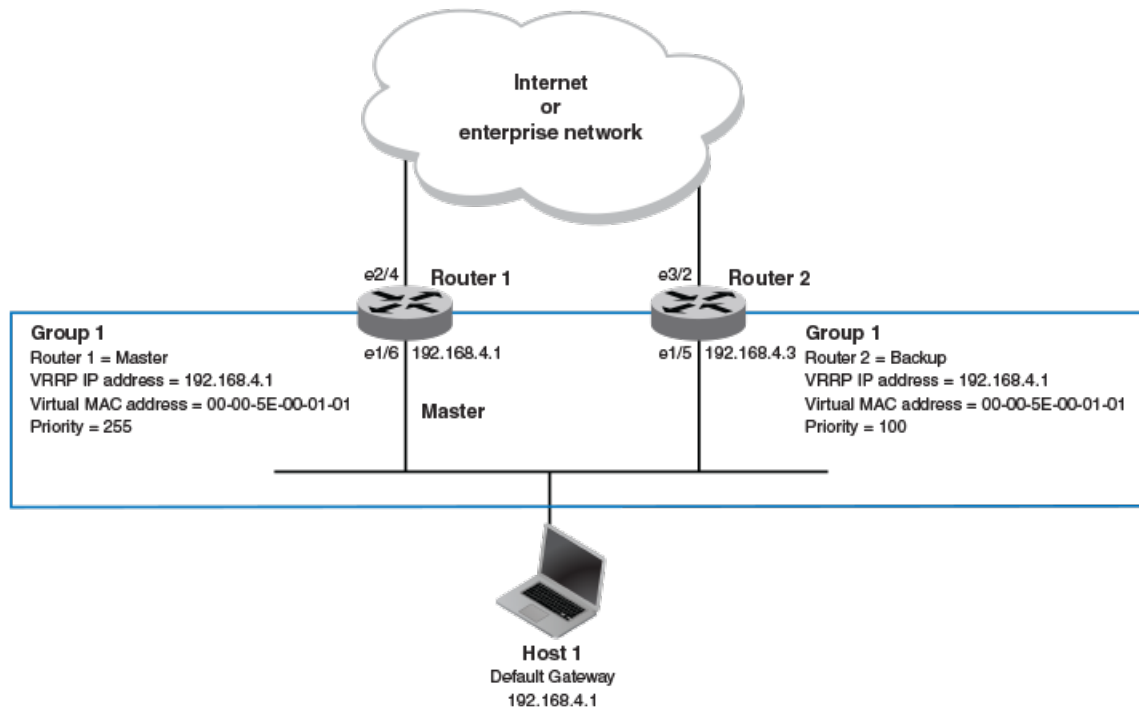
Figure 61: Devices configured as VRRP virtual routers for redundant network access for Host 1

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, OSPF, and ISIS. Their operation is unaffected when VRRP is enabled on the same interface as BGP4, OSPF, or ISIS.

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.

SLX-OS VRRP virtual router MAC address limitations

Virtual MAC (VMAC) considerations on the SLX hardware affect the number of VRRP virtual router IDs (VRIDs) available for the VRRP and VRRP Extended (VRRP-E) sessions.

When you configure a virtual routing ID (VRID), the software automatically uses the MAC address as the MAC address of the virtual router. The first five octets of the address are the standard MAC prefix for VRRP packets. The last octet is the VRID.

When the virtual router becomes the master router, it broadcasts a gratuitous ARP (GARP) request containing the virtual router's MAC address for each IP address associated with the virtual router. Hosts use the MAC address of the virtual router in routed traffic they send to their default IP gateway.

You can manually configure a unique virtual MAC address for each IPv4 and IPv6 VRRP instance per VRID. If there is no manually configured virtual MAC address for a VRRP instance, the system automatically assigns one.

Table 45: VRRP/VRRP-E VRID to VMAC mapping

Protocol	VMAC	#VMACs supported	# VRIDs supported
Standard IPv4 VRRP (v2/v3)	00-00-5E-00-01-{VRID}	16	16
Standard IPv6 VRRP (v3)	00-00-5E-00-02-{VRID}	16	16

Table 45: VRRP/VRRP-E VRID to VMAC mapping (continued)

Protocol	VMAC	#VMACs supported	# VRIDs supported
SLX-OS IPv4 VRRP-E (v2/v3)	02:E0:52:00:01:<1-byte-mvrid>	16 unique VMACs	mvrid is (((vrid-1) ÷16) + 1) with VRID ranging between 1 to 255
SLX-OS IPv6 VRRP-E (v3)	02:E0:52:00:02:<1-byte-mvrid>	16 unique VMACs	mvrid is (((vrid-1) ÷ 16) + 1) with VRID ranging between 1 to 255

> **Note**
> The 16 unique VMACs per hardware chip have to be used across all interfaces, virtual routing forwarding instances (VRFs), VRRP and VRRP-E sessions. VRIDs must not be used between VRRP/VRRP-E IPv4 and IPv6 sessions.

SLX-OS VRRP and VRRP-E interoperability

While VRRP follows RFC standards and can be used across devices from multiple vendors, Extreme Networks has its proprietary extension of VRRP, VRRP Extended (VRRP-E) which has strict interoperability rules across its devices.

In the standard SLX-OS VRRP implementation, the first five octets of the virtual MAC (VMAC) address are the standard MAC prefix for VRRP packets. The last octet is the virtual router ID (VRID). The only requirement for interoperability across other Extreme platform devices or other vendor devices is to ensure that the VRID is in the range of 1 to 16.

SLX-OS VRRP interoperability with Extreme or non-Extreme devices

The virtual router ID (VRID) must be in the range of 1 to 16.

SLX-OS VRRP-E interoperability with Extreme VDX devices

- The virtual router ID (VRID) must be in the range of 1 to 16.
- You must manually configure the VMAC on the Extreme VDX device for shared VRRP-E session with an SLX device.
- You must manually configure the VMAC on the SLX device for shared VRRP-E session with a VDX device.

SLX-OS VRRP-E interoperability with Extreme MLXe devices

- The virtual router ID (VRID) must be in the range of 1 to 16.
- You must manually configure the VMAC on the Extreme MLXe device using the MAC address from the SLX device VMAC for shared VRRP-E session with an SLX device.
- You must manually configure the VMAC on the SLX device for shared VRRP-E session with an Extreme MLXe device.

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.

VRRP authentication

The VRRP authentication type is not a parameter specific to the virtual router. VRRP uses the authentication type associated with the interfaces on which the virtual router is defined.

If your interfaces do not use authentication, neither does VRRP. For example, if you configure your device interfaces to use an MD5 password to authenticate traffic, VRRP uses the same MD5 password, and VRRP packets that do not contain the password are dropped.

In summary, if the interfaces on which you configure the virtual router use authentication, the VRRP or VRRP Extended (VRRP-E) packets on those interfaces must use the same authentication. The following VRRP and VRRP-E authentication types are supported:

- No authentication—The interfaces do not use authentication. This authentication type is the default for VRRP and VRRP-E.
- MD5—This method of authentication ensures that the packet is authentic and cannot be modified in transit. Syslog and SNMP traps are generated when a packet is dropped due to MD5 authentication failure. MD5 authentication is supported only in VRRP-E, and the device configuration is unique on a per-interface basis. The MD5 authentication configuration on an interface takes effect for all VRRP-E virtual routers configured on a particular interface.

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 30 seconds by the VRRP-E master device because VRRP-E control packets do not use the virtual MAC address.

Enabling a master VRRP device

About This Task

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.
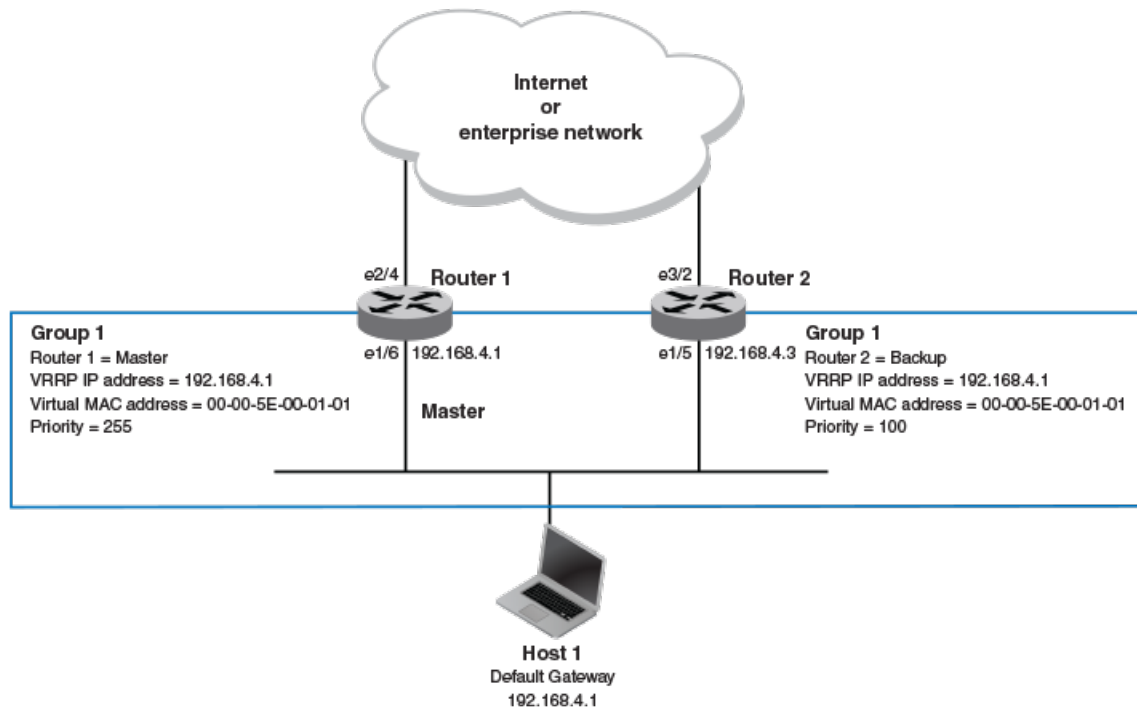
Figure 62: Basic VRRP topology

Procedure

1. On the device designated as the master VRRP device, and from privileged EXEC mode, enter configuration mode.

   ```
   device# configure terminal
   ```

2. Globally enable VRRP.

   ```
   device(config)# protocol vrrp
   ```

3. Configure the Ethernet interface link for Router 1.

   ```
   device(config)# interface ethernet 1/6
   ```

4. Configure the IP address of the interface.

   ```
   device(conf-if-eth-1/6)# ip address 192.168.4.1/24
   ```

5. Assign Router 1 to a group called Group 1.

   ```
   device(conf-if-eth-1/6)# vrrp-group 1
   ```

6. 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.

Example

The following example configures a VRRP master device.

```
device# configure
device(config)# protocol vrrp
device(config)# interface ethernet 1/6
device(conf-if-eth-1/6)# ip address 192.168.4.1/24
device(conf-if-eth-1/6)# vrrp-group 1
device(config-vrrp-group-1)# virtual-ip 192.168.4.1
```

Enabling a backup VRRP device

About This Task

This task is performed on a device that is to be designated as a backup VRRP device. For example, Router 2 in Figure 62 on page 483 is assigned as a backup device. Repeat this task for all devices that are to be designated as backup devices.

Procedure

1. On the device designated as a backup VRRP device, and from privileged EXEC mode, enter global configuration mode.

   ```
   device# configure terminal
   ```

2. Globally enable VRRP.

   ```
   device(config)# protocol vrrp
   ```

3. Configure the Ethernet interface for Router 2.

   ```
   device(config)# interface ethernet 1/5
   ```

4. Configure the IP address of interface:

   ```
   device(config-if-eth-1/5)# ip address 192.168.4.3/24
   ```

   > **Note**
   > This router will become the backup router to Router 1.

5. Assign Router 2 to the same VRRP group as Router 1.

   ```
   device(config-if-eth-1/5)# vrrp-group 1
   ```

6. 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
   ```

Example

The following example configures a backup VRRP device.

```
device# configure
device(config)# protocol vrrp
device(config)# interface ethernet 1/5
```

```
device(config-if-eth-1/5)# ip address 192.168.4.3/24
device(config-if-eth-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.*
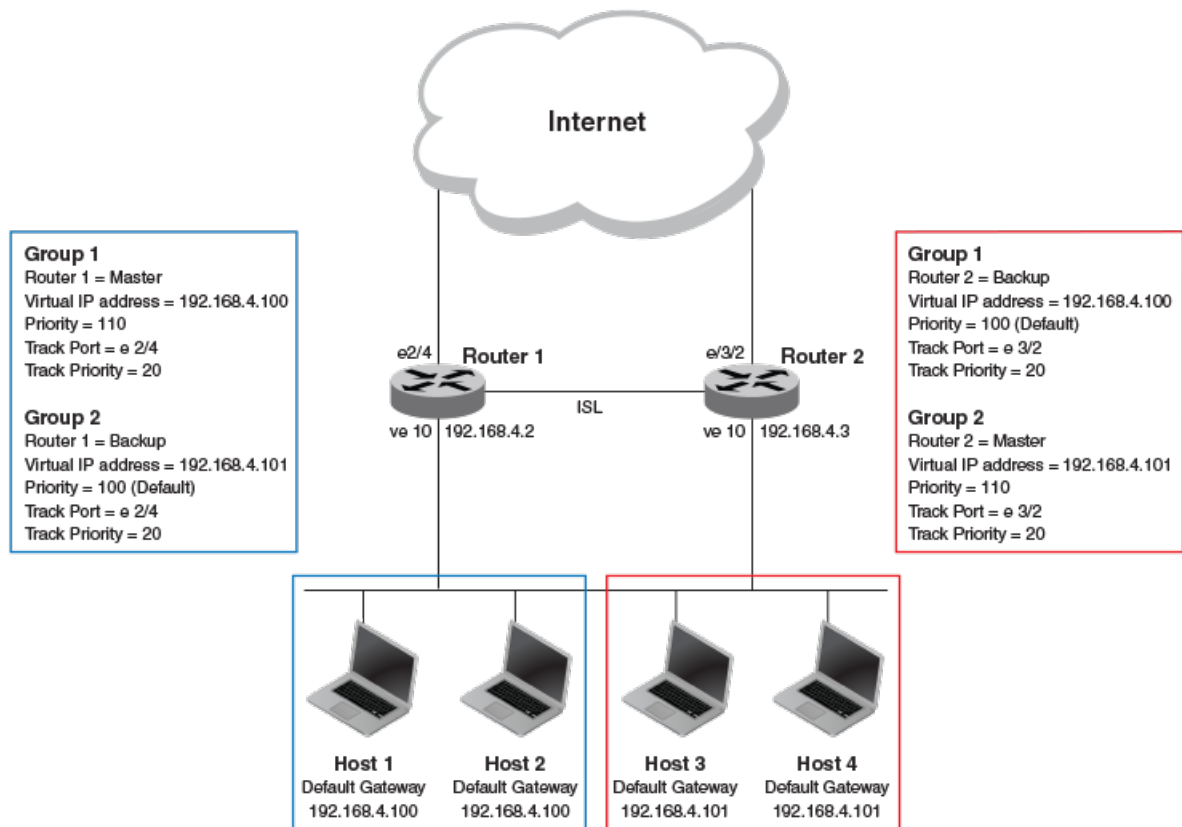
Figure 63: Two routers configured for dual redundant network access for the host

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 You Begin

Before configuring this task, ensure that a virtual LAN, named vlan 10, has been created.

About This Task

To implement the configuration of VRRP multigroup clusters as shown in Figure 63 on page 485, 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.

Procedure

1. Enable the VRRP-E protocol globally.

```
device(config)# protocol vrrp-extended
```

2. Create a VLAN.

```
device(config)# vlan 10
```

3. Bind the VLAN to the virtual Ethernet (ve) interface link for Router 1.

```
device(config-vlan 10)# router-interface ve 10
```

4. Enter virtual Ethernet (ve) interface 10.

```
device(config-vlan 10)# interface ve 10
```

5. Configure the IP address of the ve link for Router 1.

```
device(config-if-Ve-10)# ip address 192.168.4.2/24
```

6. To assign Router 1 to a VRRP-E group called Group 1, enter the command:

```
device(config-if-Ve-10)# vrrp-extended-group 1
```

7. Configure the ethernet port 2/4 as the tracking port for the interface ve 10, with a track priority of 20.

```
device(config-vrrp-extended-group-1)# track ethernet 2/4 priority 20
```

8. 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.

9. 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)# protocol vrrp-extended
device(config)# vlan 10
device(config-vlan-10)# router-interface ve 10
device(config-vlan-10)# interface ve 10
device(config-if-Ve-10)# ip address 192.168.4.2/24
device(config-if-Ve-10)# vrrp-extended-group 2
device(config-vrrp-extended-group-1)# track ethernet 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)# protocol vrrp-extended
device(config)# vlan 10
device(config-vlan-10)# router-interface ve 10
device(config-vlan-10)# interface ve 10
device(config-if-Ve-10)# ip address 192.168.4.3/24
device(config-if-Ve-10)# vrrp-extended-group 2
device(config-vrrp-extended-group-2)# track ethernet 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)# protocol vrrp-extended
device(config)# vlan 10
device(config-vlan-10)# router-interface ve 10
device(config-vlan-10)# interface ve 10
device(config-if-Ve-10)# ip address 192.168.4.3/24
device(config-if-Ve-10)# vrrp-extended-group 1
device(config-vrrp-extended-group-1)# track ethernet 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.

About This Task

In the following task steps, interface ethernet 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.
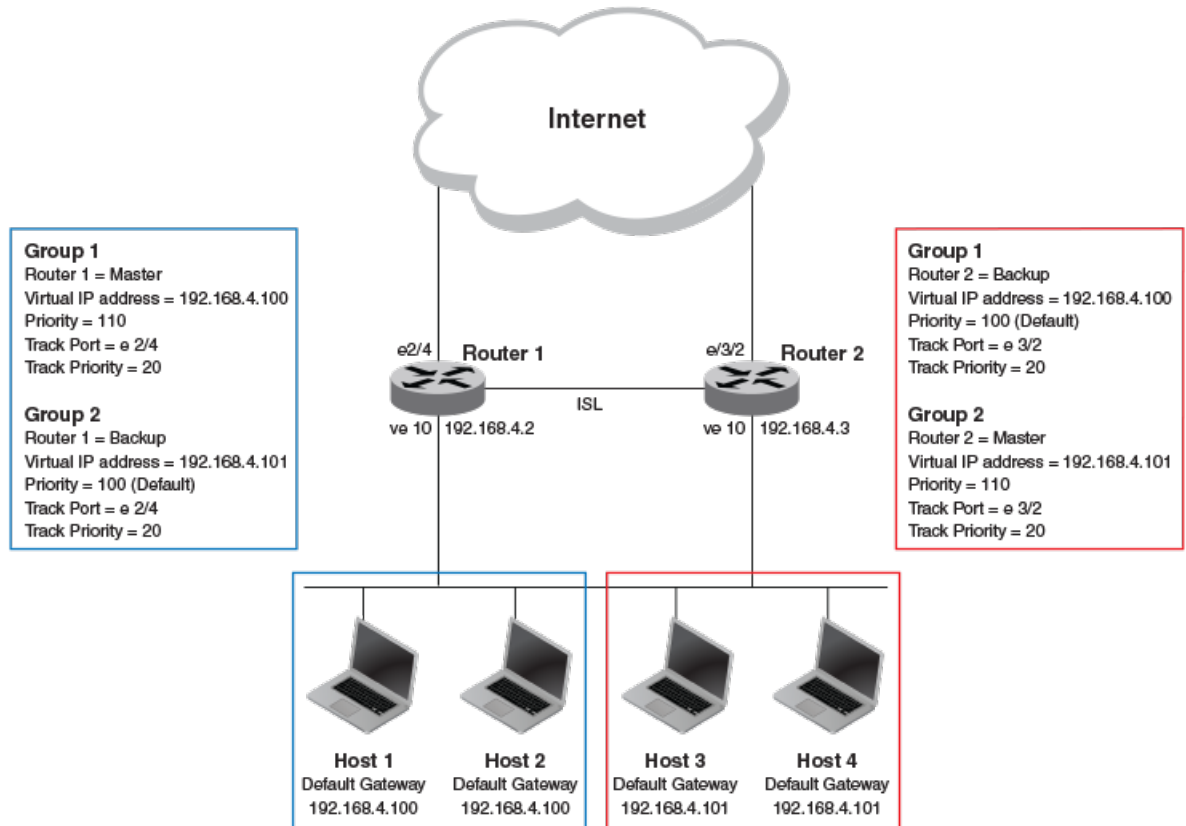
Figure 64: Multigroup VRRP routing topology

Procedure

1. Enable VRRP globally.

   ```
   device(config)# protocol vrrp
   ```

2. Enter interface configuration mode and run the following command:

   ```
   device(config)# interface ve 10
   ```

3. Run the following command to enter group configuration mode.

   ```
   device(config-if-Ve-10)# vrrp-group 1
   ```

4. Enter the **track** command to set the track port and priority:

   ```
   device(config-vrrp-group-1)# track ethernet 2/4 priority 20
   ```

Example

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)# protocol vrrp
device(config)# interface ve 10
device(config-if-Ve-10)# vrrp-group 2
device(config-vrrp-group-1)# track ethernet 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.

Before You Begin

A VRRP session must be globally enabled using the **protocol vrrp** command in global configuration mode.

About This Task

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.

Procedure

1. From global configuration mode, configure the ethernet interface for the device.

   ```
   device(config)# interface ethernet 1/5
   ```

2. Configure the IP address of the interface:

   ```
   device(conf-if-eth-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-eth-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.

Example

The following example enables preemption on a backup VRRP device.

```
device(config)# interface ethernet 1/5
device(conf-if-eth-1/5)# ip address 192.168.4.3/24
device(conf-if-eth-1/5)# vrrp-group 1
device(config-vrrp-group-1)# preempt-mode
```

Accept mode for backup VRRP devices

Accept mode allows a backup VRRP device to respond to ping, traceroute, and Telnet packets if the
backup device becomes the master VRRP device.

For each VRRP virtual routing instance, there is one master device and all other devices are backups.
Accept mode allows some network management functionality for backup VRRP devices, providing the
ability to respond to ping, traceroute, and Telnet packets. Troubleshooting network connections to the
VRRP nonowner master device is difficult unless accept mode is enabled. By default, accept mode is
enabled and non-owner VRRP devices will accept packets destined for the IPv4 or IPv6 VRID addresses
if they become master.

> **Note**
> The accept mode functionality enables a VRRP nonowner master device to respond to ping,
> Telnet, and traceroute packets, but the device will not respond to SSH packets.

Disabling accept mode on a backup VRRP device

When accept mode is disabled on a backup VRRP device, it will not respond to ping, traceroute, and
Telnet packets if the backup device becomes the master VRRP device.

About This Task

This task is performed on any device that is designated as a backup VRRP device, and the functionality
is activated if the backup device becomes a master VRRP device. Repeat this task for all devices that
are to be designated as backup devices.

> **Note**
> The accept mode functionality does not support SSH packets.

Procedure

1. On the device designated as a backup VRRP device, from privileged EXEC mode, enter global
   configuration mode.

```
device# configure terminal
```

2. Disable VRRP accept mode for all instances of VRRP on the device.

```
device(config)# vrrp-acceptmode-disable
```

You can re-enable accept mode by using the **no** form of this command.

Virtual router MAC address

When you configure a virtual routing ID (VRID), the software automatically uses the MAC address as the MAC address of the virtual router. The first five octets of the address are the standard MAC prefix for VRRP packets. The last octet is the VRID.

When the virtual router becomes the master router, it broadcasts a gratuitous ARP (GARP) request containing the virtual router's MAC address for each IP address associated with the virtual router. Hosts use the MAC address of the virtual router in routed traffic they send to their default IP gateway.

You can manually configure a unique virtual MAC address for each IPv4 and IPv6 VRRP instance per VRID. If there is no manually configured virtual MAC address for a VRRP instance, the system automatically assigns one.

The ability to configure a unique virtual MAC address is subject to the following limitations:

- This feature does not support configurable VRRP virtual MAC addresses over Multi-Chassis Trunking (MCT).
- This feature has no impact on short-path forwarding for VRRP-E.

> **Note**
> A virtual MAC address can be dynamically updated while a VRRP or VRRP-E session is enabled. When the VRRP or VRRP-E virtual MAC address is modified on the master device, expect a traffic drop until the host device receives the GARP or Router Advertisement (RA) containing the updated virtual MAC address from the master VRRP device.

Configuring unique virtual MAC addresses per VRID

In addition to system-configured standards-based virtual MAC addresses, you can manually configure a unique virtual MAC address for each IPv4 and IPv6 VRRP or VRRP-E instance per virtual routing ID (VRID). If there is no manually configured virtual MAC address (VMAC) for a VRRP instance, the system automatically assigns one.

About This Task

On SLX-OS devices, you can configure a maximum of 16 virtual router MAC addresses per device; this includes both IPv4 VRRP-E & IPv6 VRRP-E sessions.

> **Note**
> System-assigned virtual MAC addresses and manually configured virtual MAC addresses can exist at the same time on the device under the same VRID, however the configured value takes precedence. When the configured value is deleted, the assigned value again applies.

> **Note**
> A virtual MAC address can be dynamically updated while a VRRP or VRRP-E session is enabled. When the VRRP or VRRP-E virtual MAC address is modified on the master device, expect a traffic drop until the host device receives the GARP request or Router Advertisement (RA) containing the updated virtual MAC address from the master VRRP device.

To configure a unique VRRP or VRRP-E virtual MAC address for a VRID, complete the following steps.

Procedure

1. On the device designated as a VRRP-E device, from privileged EXEC mode, enter configuration mode by issuing the **configure terminal** command.

```
device# configure terminal
```

2. Globally enable the VRRP-E protocol.

```
device(config)# protocol vrrp-extended
```

3. Configure a virtual ethernet interface link.

```
device(config)# interface ve 10
```

4. Configure the IP address of the interface. All devices configured for the same VRID must be on the same subnet.

```
device(config-if-Ve-10)# ip address 10.53.5.3/24
```

5. Assign the device to VRID group 12.

```
device(config-if-Ve-10)# vrrp-extended-group 12
```

> **Note**
> You can assign a VRID number in the range of 1 through 16.

6. Manually configure an IPv4 virtual MAC address for virtual router group 12.

```
device(config-vrrp-extended-group-12)# virtual-mac 02e0.5200.0012
```

> **Note**
> System-assigned virtual MAC addresses and manually configured virtual MAC addresses can exist at the same time on the device under the same VRID, however the configured value takes precedence. When the configured value is deleted, the assigned value again applies.

7. To display IPv4 VRRP-E virtual MAC address configuration information about VRID 12 (for example), enter the following command:

```
device# show vrrp detail

Total number of VRRP session(s)   : 1

VRID 12
  Interface: Ve 10;  Ifindex: 1207959562
```

```
   Mode: VRRPE
   Admin Status: Enabled
   Description :
   Address family: IPv4
   Version: 2
   Authentication type: MD5 Authentication
   State: Initialize
   Session Master IP Address:
   Virtual IP(s): 192.168.4.100
   Virtual MAC Address: 02e0.5200.0112
 .
 .
 .
```

The partial output shows the manually configured VMAC address.

Example

The following example configures an IPv4 virtual MAC address for VRID 12 on a VRRP-E device.

```
device# configure terminal
device(config)# protocol vrrp-extended
device(config)# interface ve 10
device(config-if-Ve-10)# ip address 10.53.5.3/24
device(config-if-Ve-10)# vrrp-extended-group 12
device(config-vrrp-extended-group-12)# virtual-mac 02e0.5200.0012
```

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

About This Task

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.

Procedure

1. From privileged EXEC mode, enter configuration mode by issuing the **configure terminal** command.

   ```
   device# configure terminal
   ```

2. Globally enable VRRP-E.

   ```
   device(config)# protocol vrrp-extended
   ```

3. Configure the Virtual Ethernet (VE) interface link for the VRRP-E device.

   ```
   device(config)# interface ve 10
   ```

   Only ve interfaces are supported by VRRP-E.

4. Configure the IP address of the interface.

   ```
   device(config-if-Ve-10)# ip address 192.168.4.1/24
   ```

5. Assign the device to a group called Group 1.

   ```
   device(config-if-Ve-10)# vrrp-extended-group 1
   ```

6. 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.

7. 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)# protocol vrrp-extended
device(config)# interface ve 10
device(config-if-Ve-10)# ip address 192.168.4.1/24
device(config-if-Ve-10)# vrrp-extended-group 1
```

```
device(config-vrrp-group-1)# 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)# protocol vrrp-extended
device(config)# interface ve 10
device(config-if-Ve-10)# ip address 192.168.4.3/24
device(config-if-Ve-10)# vrrp-extended-group 1
device(config-vrrp-group-1)# priority 80
device(config-vrrp-group-1)# virtual-ip 192.168.4.100
```

Configuring MD5 authentication on IPv4 VRRP-E interfaces

Interfaces can be configured with an MD5 encrypted password for authentication, and VRRP-E can use the same authentication type associated with the interfaces on which you define the virtual router.

Before You Begin

VRRP Extended (VRRP-E) must be configured on the device and the interface associated with a virtual router group.

About This Task

Any VRRP-E packets that do not contain the password are dropped. If your interfaces do not use authentication, neither does VRRP-E. Repeat this task on all interfaces on all devices that support the same virtual router group.

> **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.

Procedure

1. From privileged EXEC mode, enter global configuration mode.

   ```
   device# configure terminal
   ```

2. Globally enable VRRP-E.

   ```
   device(config)# protocol vrrp-extended
   ```

3. Configure the Virtual Ethernet (VE) interface link for the VRRP-E device.

   ```
   device(config)# interface ve 10
   ```

   Only ve interfaces are supported by VRRP-E.

4. Enter the MD5 password configuration using the **`ip vrrp-extended auth-type`** command with a text password. The password will be encrypted when saved in the configuration file.

```
device(config-if-Ve-10)# ip vrrp-extended auth-type md5-auth kfhb61qp
```

5. Exit to privileged EXEC mode.

```
device(config-if-Ve-10)# end
```

6. Display the VRRP-E configuration to verify that MD5 authentication is enabled.

```
device# show vrrp

Total number of VRRP session(s)   : 1

VRID 1
  Interface: Ve 10;  Ifindex: 1207959562
  Mode: VRRPE
  Admin Status: Enabled
  Description :
  Address family: IPv4
  Version: 2
  Authentication type: MD5 Authentication
  State: Initialize
  Session Master IP Address:
  Virtual IP(s): 192.168.4.100
  Configured Priority: 110 (default: 100); Current Priority: unset
  Advertisement interval: 1 sec  (default: 1 sec)
  Preempt mode: DISABLE  (default: DISABLED)
  Advertise-backup: DISABLE  (default: DISABLED)
  Backup Advertisement interval: 60 sec  (default: 60 sec)
  Short-path-forwarding: Disabled
  Revert Priority: unset; SPF reverted: No
  Hold time: 0 sec  (default: 0 sec)
  Trackport:
    Port(s)                    Priority  Port Status
    =======                    ========  ===========
  Statistics:
    Advertisements: Rx: 0, Tx: 0
    Gratuitous ARP: Tx: 0
```

Example

The following example configures MD5 authentication for the specified VRRP-E interface.

```
device# configure terminal
device(config)# protocol vrrp-extended
device(config)# interface ve 10
device(config-if-Ve-10)# ip vrrp-extended auth-type md5-auth kfhb61qp
device(config-if-Ve-10)# end
device# show vrrp
```

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.

About This Task
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.

Procedure

1. Enable VRRP-E globally.

   ```
   device(config)# protocol vrrp-extended
   ```

2. Enter interface configuration mode.

   ```
   device(config)# interface ve 100
   ```

3. Run the following command to enter group configuration mode.

   ```
   device(config-if-Ve-100)# vrrp-extended-group 1
   ```

4. 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
   ```

5. Return to privileged EXEC mode.

   ```
   device(config-vrrp-group-1)# end
   ```

6. To view tracked networks with their priority and status, enter the following command:

   ```
   device# show vrrp detail

   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
.
.
.
```

Example

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)# protocol vrrp-extended
device(config)# interface ve 100
device(config-if-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.

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.
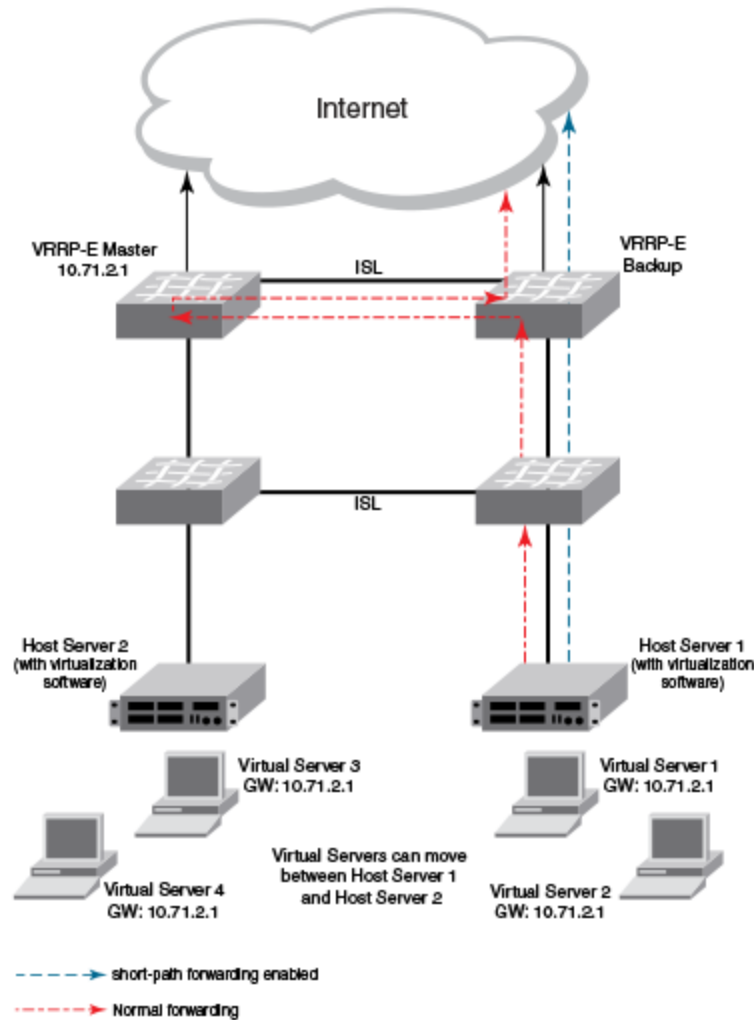
Figure 65: Short-path forwarding

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 You Begin

Before configuring VRRP-E load-balancing, VRRP-E must be configured on all devices in the VRRP-E session.

About This Task

Perform this task on all backup VRRP-E Layer 3 devices to allow load sharing within a VRRP extended group.

Procedure

1. From privileged EXEC mode, enter global configuration mode.

   ```
   device# configure terminal
   ```

2. Globally enable VRRP-E.

   ```
   device(config)# protocol vrrp-extended
   ```

3. Enter the **interface ve** command with an associated VLAN number.

   ```
   device(config)# interface ve 2019
   ```

   In this example, virtual Ethernet (ve) configuration mode is entered and the interface is assigned with a VLAN number of 2019.

4. Enter an IP address for the interface using the **ip address** command.

   ```
   device(config-ve-2019)# ip address 192.168.4.1/24
   ```

5. 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.

6. 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.

Example

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)# protocol vrrp-extended
```

```
device(config)# 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 You Begin

Before displaying VRRP information, VRRPv2 must be configured and enabled in your VRRP or VRRP-E network to generate traffic.

About This Task

Use one or more of the following commands to display VRRPv2 information. The commands do not have to be entered in this order.

Procedure

1. Enter the **show vrrp** command with a virtual-group ID to display detailed information about one virtual group ID.

```
device# show vrrp 1

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

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

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.

Before You Begin

Ensure that VRRPv2 or VRRP-Ev2 is configured and enabled in your network.

About This Task

To determine the effect of clearing the VRRP statistics, an appropriate **show** command is entered before and after the **clear** command.

Procedure

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

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

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.

Enabling IPv6 VRRPv3

IPv6 VRRPv3 is enabled on a device when a virtual IPv6 address is assigned to a VRRPv3 group.

About This Task
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.

Procedure

1. Enter the **configure** command to access global configuration mode.

   ```
   device# configure
   ```

2. To globally enable VRRPv3, enter the **ipv6 protocol vrrp** command.

   ```
   device(config)# ipv6 protocol vrrp
   ```

3. Enter the **interface ve** command with an associated VLAN number.

   ```
   device(config)# 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.

4. Enter an IPv6 address for the interface using the **ipv6 address** command.

   ```
   device(config-ve-2018)# ipv6 address 2001:2018:8192::125/64
   ```

5. 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.

6. 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.

7. 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.

Example

The following example shows how to enable a VRRPv3 session by assigning virtual IP addresses to a VRRPv3 virtual group.

```
device# configure
device(config)# ipv6 protocol vrrp
device(config)# 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.

About This Task

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.

Procedure

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

```
device# configure terminal
```

2. To globally enable VRRP, enter the **protocol vrrp** command.

```
device(config)# protocol vrrp
```

3. Configure the Ethernet interface.

```
device(config)# interface ethernet 1/6
```

In this example, Ethernet interface configuration mode is entered and the interface is assigned with a slot/port number of 1/6.

4. Enter an IPv4 address for the interface using the **ip address** command.

```
device(conf-if-eth-1/6)# ip address 192.168.5.2/24
```

5. Enter the **vrrp-group** command with a number to assign a virtual router group to the device and a version to configure VRRPv3.

```
device(conf-if-eth-1/6)# vrrp-group 10 version 3
```

In this example, a VRRPv3 group is assigned and VRRP group configuration mode is entered.

6. 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.

7. 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.

Example

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)# protocol vrrp
device(config)# interface ethernet 1/6
device(config-if-eth-1/6)# ip address 192.168.5.2/24
device(config-if-eth-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.

About This Task
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.

Procedure

1. Enter the **configure** command to access global configuration mode.

   ```
   device# configure
   ```

2. To globally enable VRRPv3, enter the **ipv6 protocol vrrp** command.

   ```
   device(config)# ipv6 protocol vrrp
   ```

3. Enter the **interface ve** command with an associated VLAN number.

   ```
   device(config)# 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.

4. Enter an IPv6 address for the interface using the **ipv6 address** command.

   ```
   device(config-ve-2018)# ipv6 address 2001:2018:8192::125/64
   ```

5. 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.

6. 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.

7. 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 ethernet 1/5 priority 15
```

8. 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.

9. 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
```

Example

The following example shows how to configure an IPv6 VRRPv3 session and enable the tracking of a 10 GbE interface.

```
device# configure
device(config)# ipv6 protocol vrrp
device(config)# 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)# track ethernet 1/5 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.

About This Task

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.

Procedure

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

   ```
   device# configure terminal
   ```

2. Enable IPv6 VRRP-E.

   ```
   device(config)# 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.

3. Enter the **interface ve** command with an associated vlan number.

   ```
   device(config)# 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.

4. Enter an IPv6 address for the interface using the **ipv6 address** command.

   ```
   device(config-ve-2018)# ipv6 address 2001:2018:8192::122/64
   ```

5. 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.

6. Enter the **description** command to enter text that describes the virtual router group.

   ```
   device(config-vrrp-group-18)# description Product Marketing group
   ```

7. 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.

8. Enter the **hold-time** command with a number representing seconds.

   ```
   device(config-vrrp-group-18)# hold-time 5
   ```

Example

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)# ipv6 protocol vrrp-extended
device(config)# 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
```

Accept mode for backup VRRP devices

Accept mode allows a backup VRRP device to respond to ping, traceroute, and Telnet packets if the backup device becomes the master VRRP device.

For each VRRP virtual routing instance, there is one master device and all other devices are backups. Accept mode allows some network management functionality for backup VRRP devices, providing the ability to respond to ping, traceroute, and Telnet packets. Troubleshooting network connections to the VRRP nonowner master device is difficult unless accept mode is enabled. By default, accept mode is enabled and non-owner VRRP devices will accept packets destined for the IPv4 or IPv6 VRID addresses if they become master.

Note
The accept mode functionality enables a VRRP nonowner master device to respond to ping, Telnet, and traceroute packets, but the device will not respond to SSH packets.

Disabling accept mode on a backup VRRP device

When accept mode is disabled on a backup VRRP device, it will not respond to ping, traceroute, and Telnet packets if the backup device becomes the master VRRP device.

About This Task

This task is performed on any device that is designated as a backup VRRP device, and the functionality is activated if the backup device becomes a master VRRP device. Repeat this task for all devices that are to be designated as backup devices.

Note
The accept mode functionality does not support SSH packets.

Procedure

1. On the device designated as a backup VRRP device, from privileged EXEC mode, enter global configuration mode.

   ```
   device# configure terminal
   ```

2. Disable VRRP accept mode for all instances of VRRP on the device.

   ```
   device(config)# vrrp-acceptmode-disable
   ```

   You can re-enable accept mode by using the **no** form of this command.

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.

About This Task

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.

Procedure

1. Enter the **configure terminal** command to enter global configuration mode.

    ```
    device# configure terminal
    ```

2. To enable VRRP globally enter the **protocol vrrp** command.

    ```
    device(config)# protocol vrrp
    ```

3. Enter the **interface ve** command with an associated VLAN number.

    ```
    device(config)# interface ve 2018
    ```

4. 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
    ```

5. 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
    ```

Example

The following example shows the v2 checksum computation method enabled for an VRRPv3 IPv4 session on a device.

```
device# configure terminal
device(config)# protocol vrrp
```

```
device(config)# 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.

About This Task
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.

Procedure

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

   ```
   device# configure terminal
   ```

2. Globally enable VRRP-Ev3.

   ```
   device(config)# ipv6 protocol vrrp-extended
   ```

3. Enter the **interface ve** command with an associated VLAN number.

   ```
   device(config)# interface ve 2019
   ```

   In this example, virtual Ethernet (ve) configuration mode is entered and the interface is assigned with a VLAN number of 2019.

4. 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
   ```

Example

The following example shows how to disable VRRPv3 RA messages from interface configuration mode for a VRRP-Ev3 session.

```
device# configure
device(config)# ipv6 protocol vrrp-extended
device(config)# 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 You Begin

Before displaying statistics, VRRPv3 must be configured and enabled in your network to generate traffic.

About This Task

Use one or more of the following commands to display VRRPv3 information. The commands do not have to be entered in this order.

Procedure

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

   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

   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 by using a CLI command.

Before You Begin

Ensure that VRRPv3 is configured and enabled in your network.

Procedure

1. Enter the **end** command, if required, to return to privileged EXEC mode.

2. Enter the **clear ipv6 vrrp statistics** command.

```
device# clear ipv6 vrrp statistics
```

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.

About This Task
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.

Procedure

1. Enter the **configure** command to access the global configuration mode.

   ```
   device# configure
   ```

2. To globally enable VRRP-Ev3, enter the **ipv6 protocol vrrp-extended** command.

   ```
   device(config)# ipv6 protocol vrrp-extended
   ```

3. Enter the **interface ve** command with an associated virtual Ethernet (VE) interface number.

   ```
   device(config)# interface ve 2019
   ```

   In this example, virtual Ethernet (ve) configuration mode is entered and the interface is assigned with a VE number of 2019.

4. Enter an IPv6 address for the interface using the **ipv6 address** command.

   ```
   device(config-if-ve-2019)# ipv6 address 2001:2019:8192::122/64
   ```

5. 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.

6. 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.

7. 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.

Example

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)# ipv6 protocol vrrp-extended
device(config)# interface ve 2019
device(config-if-ve-2019)# ipv6 address 2001:2019:8192::122/64
device(config-if-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
```

What to Do Next

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.

Configuring MD5 authentication on IPv6 VRRP-Ev3 interfaces

Interfaces can be configured with an MD5 encrypted password for authentication, and VRRP-Ev3 can use the same authentication type associated with the interfaces on which you define the virtual router.

Before You Begin

VRRP Extended version 3 (VRRP-Ev3) must be configured on the device and the interface associated with a virtual router group.

About This Task

Any VRRP-Ev3 packets that do not contain the password are dropped. If your interfaces do not use authentication, neither does VRRP-Ev3. Repeat this task on all interfaces on all devices that support the same virtual router group.

> **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.

Procedure

1. From privileged EXEC mode, enter global configuration mode.

```
device# configure terminal
```

2. Globally enable IPv6 VRRP-Ev3.

```
device(config)# ipv6 protocol vrrp-extended
```

3. Configure the Virtual Ethernet (VE) interface link for the VRRP-E device.

```
device(config)# interface ve 20
```

Only ve interfaces are supported by VRRP-E.

4. Enter the MD5 password configuration using the **ipv6 vrrp-extended auth-type** command with a text password. The password will be encrypted when saved in the configuration file.

```
device(config-if-Ve-20)# ipv6 vrrp-extended auth-type md5-auth kfhb61qp
```

When an MD5 authentication password is configured on an interface, a syslog message is displayed.

5. Exit to privileged EXEC mode.

```
device(config-if-Ve-20)# end
```

6. Display the VRRP-Ev3 configuration. In this example, only partial output is displayed to verify that MD5 authentication is configured.

```
device# show ipv6 vrrp

Total number of VRRP session(s)   : 1

VRID 1
  Interface: Ve 10;  Ifindex: 1207959562
  Mode: VRRPE
  Admin Status: Enabled
  Description :
  Address family: IPv6
  Version: 2
  Authentication type: MD5 Authentication
.
.
.
```

Example

The following example configures MD5 authentication for the specified VRRP-E interface.

```
device# configure terminal
device(config)# ipv6 protocol vrrp-extended
device(config)# interface ve 20
device(config-if-Ve-20)# ipv6 vrrp-extended auth-type md5-auth kfhb61qp
device(config-if-Ve-20)# end
device# show ipv6 vrrp
```

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.

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.
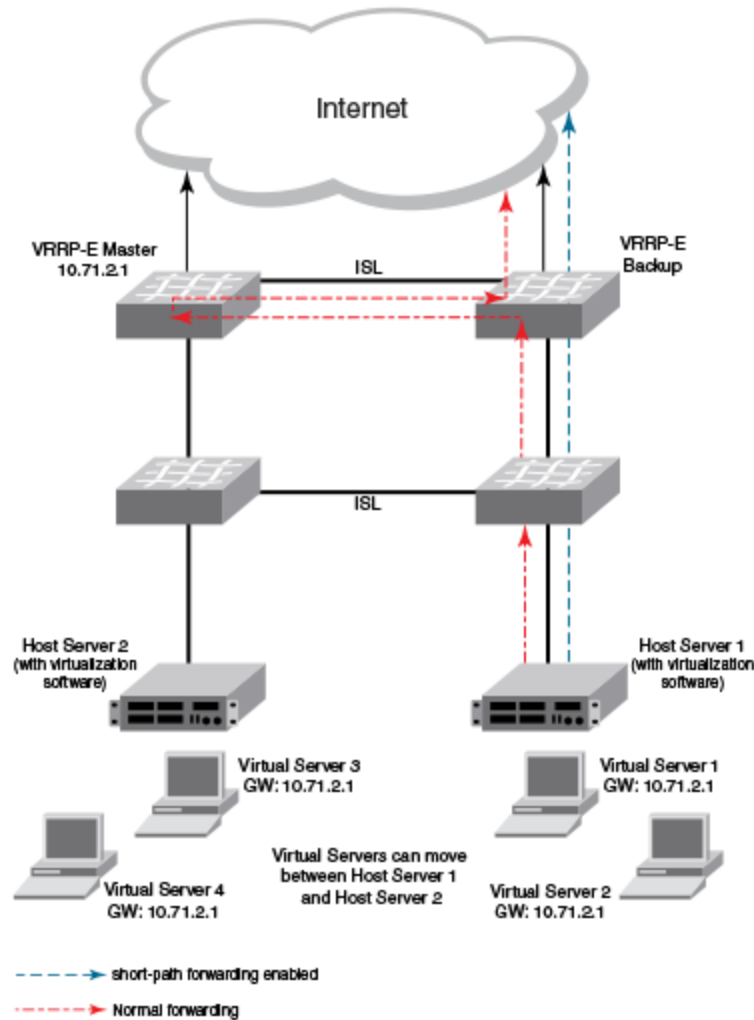
Figure 66: Short-path forwarding

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 You Begin

Before configuring VRRP-Ev3 load-balancing, VRRP-Ev3 must be configured on all devices in the VRRP-Ev3 session.

About This Task

Perform this task on all backup VRRP-Ev3 Layer 3 devices to allow load sharing within an IPv6 VRRP extended group.

Procedure

1. Use the **configure terminal** command to enter global configuration mode.

   ```
   device# configure terminal
   ```

2. Globally enable VRRP-Ev3.

   ```
   device(config)# ipv6 protocol vrrp-extended
   ```

3. Enter the **interface ve** command with an associated VLAN number.

   ```
   device(config)# interface ve 2019
   ```

   In this example, virtual Ethernet (ve) configuration mode is entered and the interface is assigned with a VLAN number of 2019.

4. Enter an IPv6 address for the interface using the **ipv6 address** command.

   ```
   device(config-ve-2019)# ipv6 address 2001:2019:8192::122/64
   ```

5. 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.

6. 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.

Example

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)# ipv6 protocol vrrp-extended
```

```
device(config)# 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 You Begin

Before displaying statistics, VRRP-Ev3 must be configured and enabled in your network to generate traffic.

About This Task

Use one or more of the following commands to display VRRP-Ev3 information. The commands do not have to be entered in this order.

Procedure

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

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. Enter the **show ipv6 vrrp 19 detail** command.

```
device# show ipv6 vrrp 19 detail

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.

4. 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
```

5. 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
```

VXLAN Layer 3 Gateway

Overview

SLX routers support VXLAN Layer 2 gateway functionality. By acting as VXLAN Layer 3 gateways, SLX routers are capable of routing Layer 3 traffic while also terminating VXLAN tunnels.

To support Layer 3 functionality, a virtual Ethernet (VE) interface must be configured over either a VLAN or a bridge domain (BD) that contains both VXLAN tunnel members and attachment circuit (AC) end-point members. Such a VE (also known as "VE over VXLAN" or "VXLAN VE") can route and switch VXLAN traffic simultaneously.

With VXLAN Layer 3 gateways over VLANs/BDs, both static/EVPN and single/logical, the following options are supported:

- Single VTEP, static VLAN
- Single VTEP, static BD
- Single VTEP, EVPN VLAN
- Single VTEP, EVPN BD
- Logical VTEP, EVPN VLAN
- Logical VTEP, EVPN BD

The following table lists and describes the support for a variety of functionalities available under VXLAN Layer 3 gateway.

Table 46: VXLAN Layer 3 gateway support for functionalities

Functionality	Description	Comments
Routing protocols	Routing protocols cannot be enabled on a VE configured as a VXLAN Layer 3 gateway.	No routing protocols (such as OSPFor ISIS) are supported on such a VE.
VRF: VRF-lite/Multi-VRF)	A VE over VXLAN can be part of a nondefault VRF.	L3VPN-VRF is not yet supported under Logical VTEP.

Table 46: VXLAN Layer 3 gateway support for functionalities (continued)

Functionality	Description	Comments
ECMP	Support ECMP paths (8) for tunnel routing.	More than eight is not restricted but is not supported. The topology must be such that more than eight paths are not present for a VXLAN tunnel.
Statistics	Tunnel statistics are supported.	By default, statistics are enabled for both directions. If hardware resources are not available, then "N/A" is displayed.
BFD	BFD is not supported.	BFD is not supported for static tunnels.
VRRP	VRRPe source IP address/EVPN-MCT is not supported.	CLI configuration is not restricted.
MTU	MTU value is not configurable.	MTU is based on an IP interface MTU. If the packet is bigger than the IP interface MTU minus the VXLAN header, the packet is dropped.
TTL	TTL value is not configurable.	Default TTL value is 64.
QoS	QoS is not configurable.	Default QoS value is 0, which is applied to the DSCP field of the IP header.
-Tunnel uniform/pipe mode for TTL/QoS	Tunnel mode (uniform or pipe mode) is not configurable	By default, tunnel mode is pipe mode: in both directions, DSCP/TTL values are not carried from or to the native packet. In the VLAN-to-VXLAN direction, the default tunnel QoS/TTL values are used.
Exporting VE-over-VXLAN interface IP address using other protocols	Routing protocols running on other IP interfaces can export the VE-over-VXLAN IP address as connected routes.	A VE with VXLAN tunnels is treated as a directly connected subnet. This VE does not support protocols, as described above. However, as there is a connected subnet, reachability to this VE can be advertised through protocols such as OSPF, ISIS, and so on, configured as part of other Layer 3 interface configurations.
Ping/Traceroute	Ping and Traceroute are supported.	Ping supports traffic from/to VXLAN tunnels.
ARP	Dynamic ARP learning is supported in the VXLAN VE.	
Proxy ARP	Proxy ARP is not supported.	Proxy ARP configuration is not restricted, but the functionality is not supported.

Table 46: VXLAN Layer 3 gateway support for functionalities (continued)

Functionality	Description	Comments
Static ARP	Static ARP is supported.	Static ARP to an IP address reachable through a VXLAN tunnel is supported. The interface in the static ARP must be configured as the VE interface to which the host on the VXLAN tunnel is connected.
IPv6	IPv6 is supported.	
Static routes	Static routes are supported.	A static route can be configured to an IP address that is reachable through a VXLAN tunnel.
RPF	Reverse path forwarding (RPF) is not supported in the VXLAN VE.	RPF configuration is not restricted, but RPF functionality is not supported.
Multicast	Layer 3 multicast is not supported.	
PBR	Policy-based routing (PBR) is not supported.	ACL/PBR for native packets is not supported.
HA/ISSU	Hitless HA/ISSU is not supported.	Traffic hits are observed.
Inter-overlay routing	• IP routing is allowed from VE over VXLAN to VLAN-VE and vice versa. • IP routing is allowed from one VE-over-VXLAN tunnel to another. • IP routing is allowed between any other tunnel type (such as GRE/IP tunnel/MPLS-based pseudowire tunnels) to a VE-over-VXLAN tunnel. • Inter-VRF routing is not supported.	
Interoperability	The Layer 3 gateway interoperates with other SLX platforms in extension mode.	Interoperability with other VXLAN-supporting devices/hypervisors is not restricted but is not supported.
CAM profile	VXLAN L2/L3 gateway is supported only in the VxlanExtended TCAM profile	The configuration is not restricted in other profiles, but functionality is not supported.
Layer 2 gateway functionality	All the present Layer 2 gateway functionalities are supported on the Layer 3 gateway.	Only static VXLAN tunnels with regular VTEPs are supported; logical VTEP tunnels are not supported.

Single-VTEP static VLAN/VXLAN-VE Layer 3 gateway

This is the most basic mode of VXLAN Layer 3 gateway (L3GW). In this mode of the L3GW, the VE is configured directly on the VLAN associated with the VXLAN Virtual Tunnel End Point (VTEP), which is mapped to the tunnel VXLAN Network Identifier (VNI).

One or more attachment circuit (AC) endpoints can be associated with the VLAN that is mapped to the VXLAN VE, or the VLAN can just contain the tunnel as its only member. The VXLAN tunnel is static, meaning that it is configured manually. The following figure illustrates a static L3GW topology.
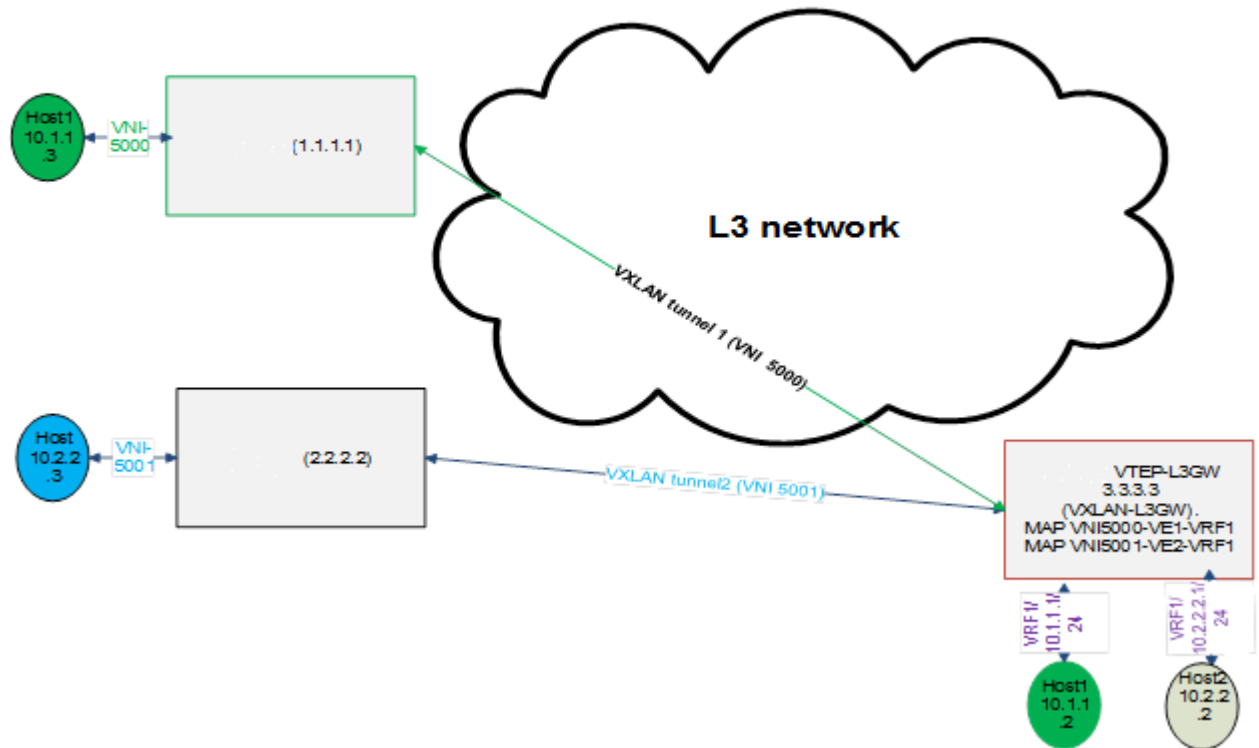

Figure 67: Static L3GW topology

The scenario for this topology is as follows:

* A customer has two subnets, 10.1.1.0/24 and 10.2.2.0/24, which have hosts 10.1.1.3 (VNI 5000) and 10.2.2.3 (VNI 5001), respectively. The VRF of the customer is configured as VRF1 on the SLX nodes.
* Also on the SLX nodes, VLAN 10 (configured with routing VE1) maps to VNI 5000, and VLAN 20 (configured with routing VE2) maps to VNI 5001.
* Host 10.1.1.2 (Ethernet 1/1, VLAN10) maps to VNI 5000, and Host 10.2.2.2 (Ethernet 1/2, VLAN 20, maps to VNI 5001).
* If Host 10.1.1.3 must communicate with Host 10.2.2.2, packets must be routed at the VRF1 level. Packets come in on VXLAN tunnel 1 (VNI 5000) and are decapsulated and sent to interface Ethernet 1/1 upon ARP resolution, or an ARP resolution is attempted if the ARP is not resolved.

Single-VTEP static BD/VXLAN-VE

This scenario is the same as for single-VTEP static VLAN/VXLAN-VE L3GW, but in this case the VE is configured on a bridge domain (BD) that is extended over the VXLAN tunnel.

EVPN-based VXLAN Layer 3 gateway

EVPN VXLAN-L3GW functionality support is one of the several key features under the larger umbrella of IP Fabrics.

The following sections describe two features for Layer 3 routing.

IP-MAC routes on a single VTEP

Similar to the normal MAC routes that are exported and installed by EVPN BGP extensions, IP-MAC routes are also exported and installed on the remote nodes. The components of this scenario are detailed here.

BGP MAC/IP routes

This kind of route represents L3-to-L2 mapping, which is basically through ARP or ND. Static, dynamic ARP/ND entries are both exported to remote PEs and get installed as host routes. IPv4/IPv6 addresses that are configured on VE interfaces are also exported.

Upon ARP learning/gleaning/snooping on a local PE, ARP/ND information is exported to its EVPN BGP peers. The information mainly includes the following: MAC, IP/IPv6, L2-VNI, and L3-VNI. Such imported ARP/ND routes are installed or withdrawn as host routes in the hardware on the remote nodes. In the control plane they are available through the ARP suppression cache, which could be further used to reply for further ARP requests from hosts attached to the remote PE.

The packet path is as follows:

- When traffic that is bound to a remote host is received on the ingress PE GW, no ARP request is generated, as the route table already has the hardware host entry to forward or route the traffic to the destined host. This situation prevents ARP flooding and further processing.
- Packets get routed on the ingress PE itself, and then are switched all the way to the destination host, through the egress PE. Because routing occurs on the ingress PE and switching occurs on the remote PE, this type of forwarding is also termed "asymmetric routing."

On the nondefault VRF, the ARP/ND exports can have two subscenarios, depending on whether L2-VNI is extended on that PE or not:

- The imported IP-MAC route can be resolved against the L2-VNI if the L2-VNI is extended over the VXLAN tunnel, in which case the packet path is similar to that described previously.
- If the L2-VNI is not extended over the tunnel on that PE, the IP-MAC route is resolved against the L3-VNI. In this case the packet path followed is similar to a prefix routes path using L3-VNI, as described in the following section.

On the default VRF, formal host IP forwarding is done.

Layer 3 VNI on a single VTEP

For multitenant scenarios using VRFs in data centers, the L3-VNI identifies a particular tenant VRF across a VXLAN-EVPN tunnel. As the name suggests, L3-VNI is used mainly for routing purposes and in short identifies the tenant VRF.

BGP IP prefix routes on VRFs are exported to the remote PE by means of EVPN (Type-5). The information mainly includes the following: Egress-PE-GW-MAC, IP/IPv6 prefix route, and L3-VNI.

Such imported IP prefix routes are imported to VRFs and are installed as VRF routes, with the VXLAN tunnel having L3-VNI as the outgoing port and remote PE-GW-MAC as the destination MAC within the inner payload L2 header.

The packet path is as follows:

- The Layer 3 routing traffic that is originated on a particular VRF is terminated on the ingress PE gateway. As part of the L3 routing within the tenant VRF on the Ingress PE, the L3 packet is carried over the VXLAN tunnel to the egress PE by means of L3-VNI. The payload packet (L3) is always marked with the egress PE as the next hop.
- When the packet arrives at the egress PE, the outer header L3-VNI is used as an identifier to the tenant VRF, and the inner packet gets routed within this tenant VRF context.
- Because routing takes place on both the ingress and egress PE, this routing is also termed "symmetric routing."

The following figure illustrates this topology.
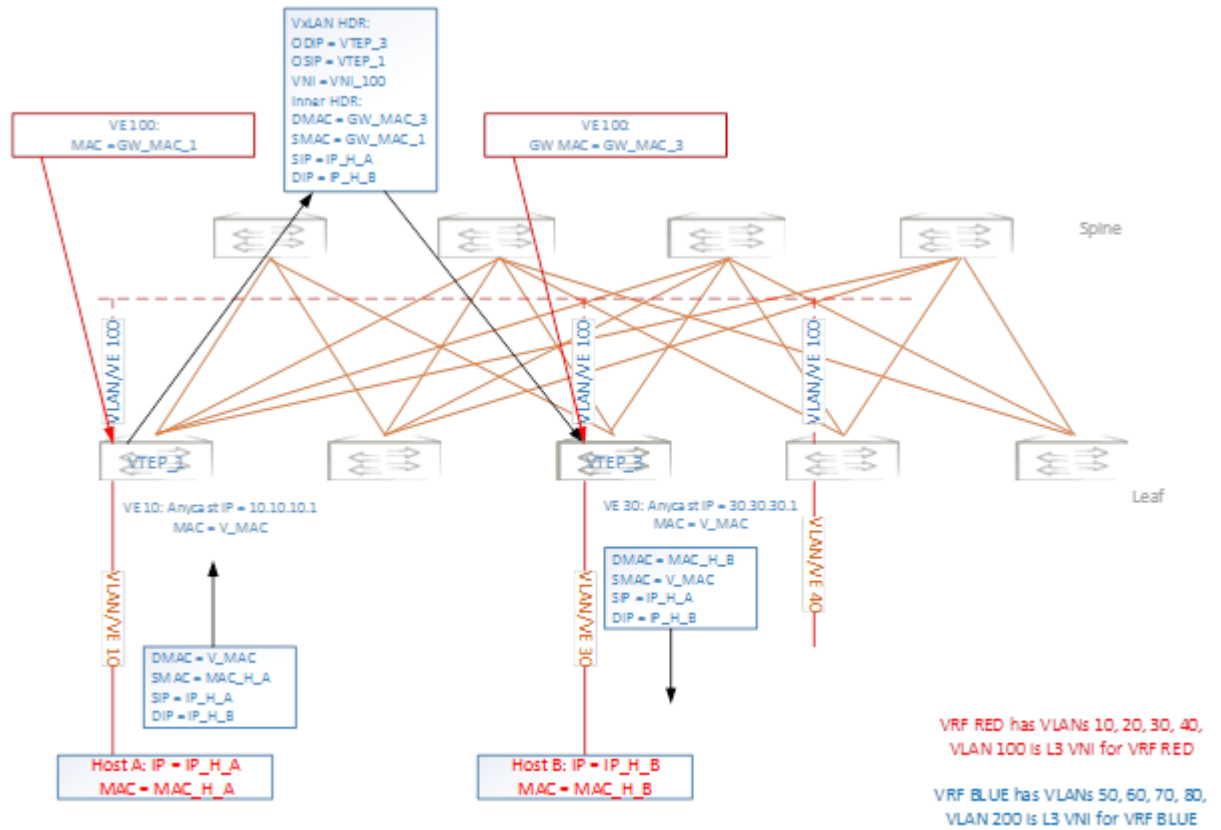
Figure 68: Layer 3 VNI on a single VTEP

EVPN-based VXLAN Layer 3 gateway on LVTEP

This section discusses the Layer 3 functionality support on such a logical VTEP (LVTEP).

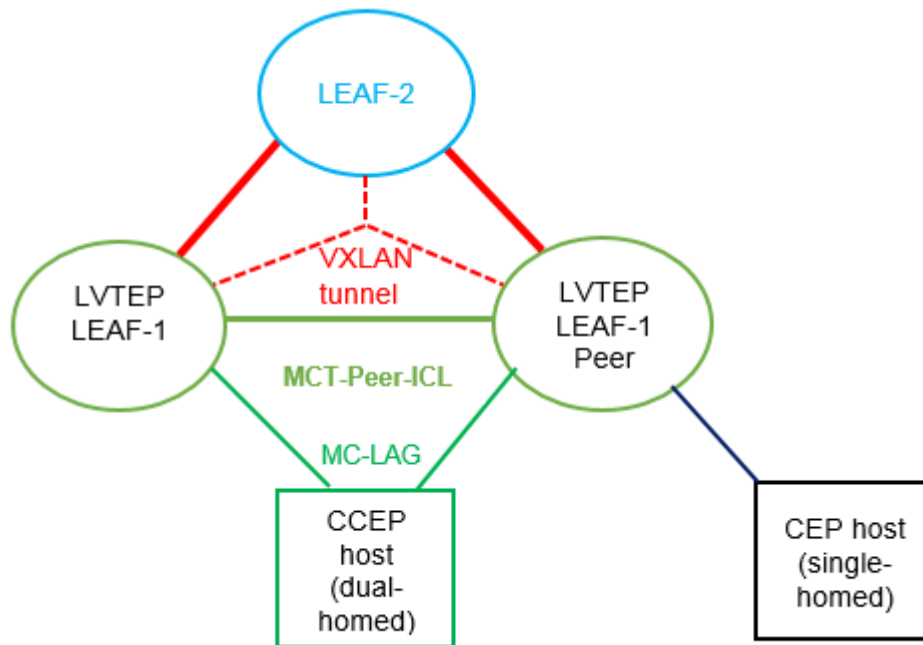The following figure illustrates a VXLAN LVTEP topology.

Figure 69: VXLAN LVTEP topology

The LVTEP is formed through MCT peering (spoke-PW-peer) between Leaf-1 and its peer node, Leaf-12-peer, to provide redundancy for a VXLAN leaf node.

A VXLAN tunnel is created between such an LVTEP leaf and a remote leaf. The source IP address of the VXLAN tunnel is the same on both nodes. Therefore, the tunnel has a single tunnel representation on the remote leaf (Leaf-2 in the figure). The logical connection of the tunnel is shown as the dotted red line.

The single tunnel on Leaf-2 has two underlay paths to reach Leaf-1 and the Leaf-1 peer. Any traffic southbound from Leaf-2 is load balanced and can end up in either of the LVTEP peers.

IP-MAC routes on LVTEP

Similar to the single-VTEP case, the normal MAC-IP routes are exported and installed by EVPN BGP extensions on LVTEP between the leaf nodes, providing for the following behavior:

* Only one of the LVTEP peers that learns ARP (source LVTEP node) exports the route to the remote leaf. Such an imported route is installed as a host route that is pointed to the VXLAN tunnel, which has two underlay paths.

* The source LVTEP also syncs the ARP route to its LVTEP peer over ICL as part of MCT. These routes are installed pointing to the ICL interface (PW). Such synced IP-MAC routes are not readvertised to the VXLAN peer.

* The remote leaf exports its IP-MAC routes to both the LVTEP peers, and both peers install the routes in hardware—as host routes pointing to the local VXLAN tunnel toward the remote leaf.

A BGP MAC/IP route represents L3-to-L2 mapping, which is basically ARP or ND. Static and dynamic ARP/ND entries are exported to remote PEs and get installed as host routes. IPv4/IPv6 addresses that are configured on VE interfaces are also exported.

Upon ARP learning/gleaning/snooping on a local PE, ARP/ND information is exported to its EVPN BGP peers. The information mainly includes the following: MAC, IP/IPv6, L2-VNI, L3-VNI, and ESI segment. (In VXLAN, the ESI segment ID is always 0.)

Such imported ARP/ND routes are installed or withdrawn as host routes in the hardware on the remote nodes. In the control plane they are available through the ARP suppression cache, which can be further used to reply for further ARP requests from hosts that are attached to the remote PE.

The packet path is as follows:

- When traffic bound to a remote host is received on the Ingress PE GW, no ARP request is generated, as the route table already has the hardware host entry to forward/route the traffic to the destined host. This situation prevents ARP flooding and further processing.

- Packets get routed on the ingress PE itself, and then are switched all the way to the destination host, through the egress PE. Because routing occurs on the ingress PE and switching on the remote PE, this type of forwarding is also termed "asymmetric routing."

On the nondefault VRF, the ARP/ND exports can have two subscenarios, depending on whether L2-VNI is extended on that PE or not:

- The imported IP-MAC route could be resolved against L2-VNI if L2-VNI is extended over the VXLAN tunnel, in which case the packet path is similar to the one described previously.

- If L2-VNI is not extended over tunnel on that PE, the IP-MAC route is resolved against L3-VNI, in which case the packet path followed is similar to the prefix routes path using L3-VNI.

On the default VRF, normal host IP forwarding always occurs.

L3-VNI on LVTEP

Similar to the single-VTEP case, the IP prefix routes are also exported and installed by EVPN BGP extensions on LVTEP between the leaf nodes (with Type-5 routes), providing for the following:

- LVTEP peers export the IP prefix routes over BGP EVPN to remote leaf peer(s). Such imported routes are installed as prefix routes pointing to the VXLAN tunnel, which has two underlay paths.

- The IP prefix routes are also synced across the LVTEP peers on an MCT-ICL link and are installed as pointing to the ICL (PW). Such synced routes are not advertised to VXLAN peers.

- The remote leaf exports its IP prefix routes to both the LVTEP peers, both of which install the routes in hardware as network/prefix routes pointing to the local VXLAN tunnel.

BGP IP prefix routes on VRFs are exported to the remote PE over EVPN (Type-5). The information mainly includes the following: Egress-PE-GW-MAC, IP/IPv6 Prefix route, L3-VNI, ESI segment. (In VXLAN, the ESI segment ID is always 0.)

Such imported IP prefix routes are imported to VRFs and installed as VRF routes, with the VXLAN tunnel having L3-VNI as the outgoing port and remote PE-GW-MAC as the destination MAC with in the Inner Payload L2 header.

The packet path is as follows:

- The L3/routing traffic originated on a particular VRF is terminated on the ingress PE gateway. As part of the L3 routing with in the tenant VRF on the ingress PE, the L3 packet is carried over the VXLAN tunnel to the egress PE over L3-VNI. The payload packet (L3) is always marked with the egress PE as the next-hop.

- When the packet arrives at the egress PE, the outer header L3-VNI is used as an identifier to the tenant VRF, and the inner packet gets routed within this tenant VRF context.
- Because routing takes place on both the ingress and egress PE, this is also termed "symmetric routing."

Configuring VXLAN Layer 3 gateway

Configure the TCAM Profile for Layer 3 Gateway

Layer 3 gateway is supported only in the default TCAM profile.

About This Task

> **Note**
> This procedure requires a system reload to activate the profile configuration.

Procedure

1. Enter global configuration mode.

   ```
   device# configure terminal
   ```

2. Enter hardware configuration mode.

   ```
   device(config)# hardware
   ```

3. Specify the TCAM profile.

   ```
   device(config-hardware)# profile tcam default
   ```

   This example runs the **profile tcam** command with the **default** parameter.

4. Access privileged EXEC mode.

   ```
   device(config-hardware)# exit
   ```

5. Save the running configuration to the startup configuration file.

   ```
   device# copy running-config startup config
   ```

6. Activate the profile configuration.

   ```
   device# reload system
   ```

Example

This example summarizes the commands in the procedure.

```
device# configure terminal
device(config)# hardware
device(config-hardware)# profile tcam default
device(config-hardware)# exit
device# copy running-config startup config
device# reload system
```

Configuring a single-VTEP static VLAN/VE Layer 3 gateway

Follow these steps to configure a single-VTEP static VLAN/VE Layer 3 gateway.

Procedure

1. Configure the TCAM profile to support L3GW. Complete the steps in Configure the TCAM Profile for Layer 3 Gateway on page 534

2. Configure a loopback address.

```
interface Loopback 100
 no shutdown
 ip address 40.40.40.1/32
```

3. Configure an IP interface to be the interface of the VXLAN tunnel.

```
interface Ethernet 0/10
 ip proxy-arp
 ip address 50.50.50.1/24
 no shutdown
```

4. Create a VLAN.

```
vlan 500
```

5. Configure an attachment circuit (AC) endpoint.

```
interface Ethernet 0/20
 switchport
 switchport mode trunk
 switchport trunk allowed vlan add 500
 switchport trunk tag native-vlan
 no shutdown
```

6. Configure the VTEP.

```
overlay-gateway test
 type layer2-extension
 ip interface Loopback 100
 map vlan 500 vni 15000
 activate
 site VCS_2
 ip address 40.40.40.2  << This must be the remote-end loopback IP address and be
reachable
 extend vlan add 500
```

7. Configure VE over VXLAN by configuring a VE over the VLAN associated with VXLAN.

```
vlan 500
 router-interface ve 500

int ve 500
 ip address 15.15.15.1/24
 ipv6 address 1001::1/64
 no shutdown
```

Configuring a single-VTEP static BD/VE Layer 3 gateway

Follow these steps to configure a single-VTEP static BD/VE Layer 3 gateway.

Procedure

1. Configure the TCAM profile to support L3GW. Complete the steps in Configure the TCAM Profile for Layer 3 Gateway on page 534.

2. Configure a loopback address.

```
interface Loopback 100
 no shutdown
 ip address 40.40.40.1/32
```

3. Configure an IP interface to be the interface of the VXLAN tunnel.

```
interface Ethernet 0/10
 ip proxy-arp
 ip address 50.50.50.1/24
 no shutdown
```

4. Create a VLAN.

```
vlan 500
```

5. Configure an attachment circuit (AC) endpoint with a logical interface.

```
interface Ethernet 0/20
 switchport
 switchport mode trunk-no-default
 logical-interface eth 0/20.500 vlan 500
 no shut
```

6. Configure a bridge domain (BD).

```
bridge-domain 500 p2mp
 logical-interface eth 0/20.500
 pw-profile default
 bpdu-drop-enable
 local-switching
```

7. Configure the VTEP.

```
overlay-gateway test
 type layer2-extension
 ip interface Loopback 100
 map vlan 500 vni 15000
 activate
 site VCS_2
 ip address 40.40.40.2  << This must be the remote-end loopback IP address and be
reachable
 extend bridge-domain add 500
```

8. Configure VE over VXLAN by configuring a VE over the VLAN that is associated with VXLAN.

```
vlan 500
 router-interface ve 500

int ve 500
 ip address 15.15.15.1/24
 ipv6 address 1001::1/64
 no shutdown
```

Configuring an EVPN Layer 3 gateway for MAC IP routes

Do the following to configure a Layer 3 gateway for MAC IP routes.

About This Task

This configuration is similar to that for Layer 2 EVPN MAC routes.

Procedure

1. Configure the TCAM profile to support L3GW. Complete the steps in Configure the TCAM Profile for Layer 3 Gateway on page 534.

2. Create VLANs to be extended on the tunnel.

```
vlan 1-4
```

3. Configure an EVPN instance.

```
evpn r1
 route-target both auto
 rd auto
 vlan add 2-4
```

4. Configure BGP.

```
router bgp
  local-as 100
    neighbor 98.0.0.1 remote-as 100
  address-family ipv4 unicast
  !
  address-family ipv6 unicast
  !
  address-family l2vpn evpn
    graceful-restart
    neighbor 98.0.0.1 encapsulation vxlan
    neighbor 98.0.0.1 activate
```

5. Configure a loopback interface.

```
interface Loopback 1
 no shutdown
 ip ospf area 0
 ip address 1.2.3.4/32
```

6. Configure a tunnel interface.

```
interface Ethernet 0/4
 ip ospf area 0
 ip proxy-arp
 ip address 98.0.0.2/24
 no shutdown
```

7. Configure the overlay gateway.

```
overlay-gateway g1
type layer2-extension
ip interface Loopback 1
  map vni auto
activate
```

Configuring an EVPN Layer 3 VNI

Follow these steps to configure an EVPN Layer 3 VNI.

About This Task

Layer 3 VNI is used to support VRFs.

Procedure

1. Configure the TCAM profile to support L3GW. Complete the steps in Configure the TCAM Profile for Layer 3 Gateway on page 534.

2. Create VLANs to be extended on the tunnel.

```
vlan 1-4
```

3. Configure an EVPN instance.

```
evpn r1
 route-target both auto
 rd auto
 vlan add 2-4
```

4. Configure BGP.

```
router bgp
  local-as 100
    neighbor 98.0.0.1 remote-as 100
  address-family ipv4 unicast
  !
  address-family ipv6 unicast
  !
  address-family l2vpn evpn
    graceful-restart
    neighbor 98.0.0.1 encapsulation vxlan
    neighbor 98.0.0.1 activate
```

5. Configure a loopback interface.

```
interface Loopback 1
 no shutdown
 ip ospf area 0
 ip address 1.2.3.4/32
```

6. Configure a tunnel interface.

```
interface Ethernet 0/4
 ip ospf area 0
 ip proxy-arp
 ip address 98.0.0.2/24
 no shutdown
```

7. Configure the overlay gateway.

```
overlay-gateway g1
type layer2-extension
ip interface Loopback 1
  map vni auto
activate
```

8. Configure a VRF.

```
vrf red
rd 5:50
evpn irb ve 100 <peer-gateway> <--Identifies the L3-VNI
address-family ipv4 unicast
  route-target export 5:100 evpn
  route-target import 5:100 evpn
!
address-family ipv6 unicast
  route-target export 5:100 evpn
  route-target import 5:100 evpn
```

9. Configure the L3-VNI Integrated Routing and Bridging (IRB) instance. (An IP address is not required.)

```
vlan 100  <-- The L3-VNI is a VLAN as a result of auto map mode
router-interface Ve 100
```

10. Alternatively, configure the L3-VNI IRB by using a BD. (The L3-VNI is 4K+BD-ID as a result of auto mapping.)

```
bridge-domain 500 p2mp
 router-interface ve 100
!
interface Ve 100
```

```
vrf forwarding red
no shutdown
```

What to Do Next

Configuring an EVPN LVTEP for MAC IP routes

Follow these steps to configure an EVPN LVTEP for MAC IP routes.

About This Task

This configuration is similar to that for Layer 2 EVPN MAC routes.

Procedure

1. Configure the TCAM profile to support L3GW. Complete the steps in Configure the TCAM Profile for Layer 3 Gateway on page 534.

2. Create VLANs to be extended on the tunnel.

```
vlan 1-4
```

3. Configure an EVPN instance.

```
evpn r1
 route-target both auto
 rd auto
 vlan add 2-4
```

4. Configure BGP.

```
router bgp
 local-as 100
  neighbor 3.3.3.3 remote-as 101 <-- The VXLAN peer
  neighbor 40.40.100.50 remote-as 102 <-- The MCT peer
 address-family ipv4 unicast
 !
 address-family ipv6 unicast
 !
 address-family l2vpn evpn
  graceful-restart
  neighbor 3.3.3.3 encapsulation vxlan <-- The VXLAN peer
  neighbor 3.3.3.3 activate
  neighbor 40.40.100.50 encapsulation mpls <-- The MCT peer
  neighbor 40.40.100.50 activate
```

5. Configure a loopback interface for BGP neighborship.

```
interface Loopback 1
 no shutdown
 ip ospf area 0
 ip address 40.40.100.40/32
```

6. Configure a loopback interface for VXLAN. (MCT peers must have the same address.)

```
interface Loopback 2
 no shutdown
 ip ospf area 0
 ip address 2.2.2.2/32
```

7. Configure a tunnel interface.

```
interface Ethernet 0/4
 ip ospf area 0
 ip proxy-arp
 ip address 98.0.0.2/24
 no shutdown
```

8. Configure the overlay gateway.

```
overlay-gateway g1
 type layer2-extension
 ip interface Loopback 1
  map vni auto
 activate
```

9. Configure the MCT cluster.

```
cluster c1 1
 peer-interface Ve 45
 peer 40.40.100.50
 deploy
```

Configuring an EVPN Layer 3 VNI for LVTEP

This example configures an L3-VNI to support a VRF.

Procedure

1. Configure the TCAM profile to support L3GW. Complete the steps in Configure the TCAM Profile for Layer 3 Gateway on page 534.

2. Instantiate VLANs to be extended on the tunnel.

```
vlan 1-4
```

3. Configure an EVPN instance.

```
evpn r1
 route-target both auto
 rd auto
 vlan add 2-4
```

4. Configure BGP.

```
router bgp
 local-as 100
  neighbor 3.3.3.3 remote-as 101 <-- The VXLAN peer
  neighbor 40.40.100.50 remote-as 102 <-- The MCT peer
 address-family ipv4 unicast
 !
 address-family ipv6 unicast
 !
 address-family l2vpn evpn
  graceful-restart
  neighbor 3.3.3.3 encapsulation vxlan <-- The VXLAN peer
  neighbor 3.3.3.3 activate
  neighbor 40.40.100.50 encapsulation mpls <-- The MCT peer
  neighbor 40.40.100.50 activate
 !
```

5. Configure a loopback interface for BGP neighborship.

```
interface Loopback 1
 no shutdown
 ip ospf area 0
 ip address 40.40.100.40/32
```

6. Configure a loopback interface for VXLAN. (MCT peers must have the same address.)

```
interface Loopback 2
 no shutdown
 ip ospf area 0
 ip address 2.2.2.2/32
```

7. Configure a tunnel interface.

```
interface Ethernet 0/4
 ip ospf area 0
 ip proxy-arp
 ip address 98.0.0.2/24
 no shutdown
```

8. Configure the overlay gateway.

```
overlay-gateway g1
 type layer2-extension
 ip interface Loopback 1
  map vni auto
 activate
```

9. Configure the MCT cluster.

```
cluster c1 1
 peer-interface Ve 45
 peer 40.40.100.50
 deploy
```

10. Configure the VRF.

```
vrf red
rd 5:50
evpn irb ve 100 <peer-gateway> <-- Identifies the L3-VNI
address-family ipv4 unicast
  route-target export 5:100 evpn
  route-target import 5:101 evpn  <-- From the MCT peer
  route-target import 5:102 evpn  <-- From the VXLAN peer
!
address-family ipv6 unicast
```

11. Configure the L3-VNI IRB. (An IP address is not needed.)

```
vlan 100
 router-interface Ve 100
!
interface Ve 100
 vrf forwarding red
 no shutdown
```

Example show and clear commands for VXLAN Layer 3 gateway

This section presents example output of the **show** and **clear** commands that are useful in managing VXLAN Layer 3 gateway.

show overlay-gateway

The following is example output from the **show overlay-gateway** command.

```
device# show overlay-gateway
Overlay Gateway "VXLAN", ID 1, rbridge-ids 1
Admin state up
IP address 33.32.31.13 (loopback 1), Vrf default-vrf
Number of tunnels 6
Packet count: RX 17909 TX 1247
Byte count : RX (500125) TX 356626
```

show tunnel brief

The following is example output from the **show tunnel brief** command.

```
device# show tunnel brief
Tunnel 1, mode VXLAN, rbridge-ids 1
Admin state up, Oper state up
Source IP 33.32.31.13, Vrf default-vrf
Destination IP 33.32.31.10
Tunnel 2, mode VXLAN, rbridge-ids 1
Admin state up, Oper state up
Source IP 33.32.31.13, Vrf default-vrf
Destination IP 33.32.31.1
```

show tunnel

The following is example output from the **show tunnel** *ID* command.

```
device# show tunnel 1
Tunnel 1, mode VXLAN, rbridge-ids 1
Ifindex 2080374798, Admin state up, Oper state up
Overlay gateway "VXLAN", ID 1
Source IP 33.32.31.13 (loopback 1), Vrf default-vrf
Destination IP 33.32.31.10
Active next hop on rbridge1:
IP: 33.32.31.10, Vrf: default-vrf
Egress L3 port: Ve 10, Outer SMAC: 0027.f886.bb36
Outer DMAC: e41f.1343.97d2
Egress L2 Port: Po 11, Outer ctag: 10
BUM forwarder: yes
Packet count: RX 18492 TX 1242
Byte count : RX (NA) TX 356307
```

show vlan brief

The following is example output from the **show vlan brief** command, to verify VNI mapping.

```
device# show vlan brief
Total Number of VLANs configured : 1
Total Number of VLANs provisioned : 1
Total Number of VLANs unprovisioned : 0
VLAN Name State Ports Classification
(F)-FCoE (u)-Untagged, (t)-Tagged
(R)-RSPAN (c)-Converged
(T)-TRANSPARENT
30 VLAN0030 ACTIVE Po 11(t)
Tu 1(t) vni 100
Tu 2(t) vni 100
```

show mac-address-table

The following is example output from the **show mac-address-table** command with the **bridge-domain** keyword, to verify gateway MAC addresses.

```
device# show mac-address-table bridge-domain
VlanId/BD-Id   Mac-address           Type     State      Ports/LIF/peer-ip
629(B)            0011.2222.5555     Dynamic  Active     eth 1/3.100
```

```
629(B)          0011.2222.6666    Dynamic  Inactive  eth 1/1.500
629(B)          0011.2222.1122    Dynamic  Active    10.12.12.12
629(B)          0011.2222.3333    static   Inactive  po 5.700
629(B)          0011.0101.5555    Dynamic  Active    eth 1/2.400

Total MAC addresses : 5
```

show ip arp suppression-cache

The following is example output from the **show ip arp suppression-cache** command, to verify remote ARPs and NDs..

```
device# show ip arp suppression-cache
Flags: L  - Locally Learnt Adjacency
       R  - Remote Learnt Adjacency
       RS - Remote Static Adjacency
Vlan/Bd    IP                Mac             Interface
Age         Flags
-----------------------------------------------------------------------------------
---------
0100 (V)   10.10.10.11     0000.0a0a.0a0b X/X
00:01:35        L
0101 (V)   10.10.10.98     0000.1111.0000 X/X
Never           RS
0101 (V)   10.10.10.99     609c.9f5a.4d15 X/X
Never           RS
0102 (V)   12.12.122.1     609c.9f5a.4715 X/X
Never           RS
```

show bgp evpn

All the options under the **show bgp evpn** command are helptul.

The following is example output from the **show bgp evpn l3vni** command, for a specific VRF .

```
device# show bgp evpn l3vni vrf red

-----------------------------------------------------------------------
   L3VNI Prefix Origination Conditions for vrf (red)
-----------------------------------------------------------------------
Address Family under BGP : True
RD Configured          : True
IRB I/F Configured     : True (0x48000064)
IRB I/F Status         : False
IRB EVID Conigured     : True (100)
Router mac Exists      : True
Source VTEP            : 40.40.40.1
VTEP Active            : Active
IPv4 L3VNI Active          : Active
IPv6 L3VNI Active          : Inactive


-----------------------------------------------------------------------
   L3VNI Prefix Import Conditions for vrf (red)
-----------------------------------------------------------------------
Address Family under BGP : True
IRB I/F Configured     : True (0x48000064)
IRB EVID Configured    : True (100)
Router mac Exists      : True
IPv4 L3VNI Active          : Active
```

```
IPv6 L3VNI Active             : Inactive
```

The following is example output from the **show bgp evpn routes** command, with ARP specified.

```
device# show bgp evpn routes type arp

Total number of BGP EVPN ARP Routes : 4 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
Route Distinguisher: 40.40.100.50:32869
1     ARP:[0][0000.0a0a.0a0b]:[IPv4:10.10.10.11]
                          0.0.0.0          0          100        0      BL
        AS_PATH:
          L2 Label: 101 L3 Label: 100
        ESI : 00.000000000000000000
2     ARP:[0][609c.9f5a.4715]:[IPv4:15.143.15.1]
                          0.0.0.0          0          100        0      BL
        AS_PATH:
          L2 Label: 101 L3 Label: 100
        ESI : 00.000000000000000000
Route Distinguisher: 40.40.100.50:33769
3     ARP:[0][609c.9f5a.4715]:[IPv4:14.13.15.1]
                          0.0.0.0          0          100        0      BL
        AS_PATH:
          L2 Label: 1001 L3 Label: 100
        ESI : 00.000000000000000000
Route Distinguisher: 40.40.100.60:32869
4     ARP:[0][609c.9f5a.8d15]:[IPv4:6.6.2.5]
                          40.40.40.2       0          100        0      BE
        AS_PATH: 1000
          L2 Label: 101 L3 Label: 0
        ESI : 00.000000000000000000
```

The following is example output from the **show bgp evpn routes** command, with IPv4 prefixes specified.

```
device# show bgp evpn routes type ipv4-prefix briief
Total number of BGP EVPN Ipv4Prefix Routes : 4 Status codes: s suppressed, d damped, h
history, * valid, > best, i internal, S stale Origin codes: i - IGP, e - EGP, ? -
incomplete
    Network           Next Hop        MED        LocPrf      Weight Path
Route Distinguisher: 5:50
*>  IP4Prefix:[0][14.13.15.0/24]
                          0.0.0.0          0          100        0      ?
*>  IP4Prefix:[0][15.143.15.0/24]
                          0.0.0.0          0          100        0      ?
*>  IP4Prefix:[0][16.16.16.0/24]
                          0.0.0.0          0          100        0      ?
Route Distinguisher: 5:100
*>  IP4Prefix:[0][17.17.17.0/24]
                          40.40.40.2       0          100        0      1000 ?
```

show tunnel statistics

The following is example output from the **show tunnel statistics** command.

```
device# show tunnel statistics
Tnl ID RXpackets TX packets RX bytes TX bytes
======== =============== =============== =============== ================
1      18573        1242        356307      356307
```

clear counters all

Use the **clear counters all** command to clear statistics on a gateway, including tunnel statistics.

clear overlay-gateway

Use the **clear overlay-gateway** command to clear statistics on an overlay gateway, including tunnel statistics.

QoS for VXLAN Layer 2 and Layer 3 Gateway Interconnections

SLX-OS VXLAN Layer 2 and Layer 3 gateway interconnections can support QoS.

VXLAN Layer 3 gateways resolve the ARP for host VMs and create routes. When VM1 sends inter-subnet packets to VM3, Leaf-A sends the packet to Spine-1 over the VXLAN configuration in the following figure. First, Spine-1 decapsulates the original Layer 2 frame inside the VXLAN tunnel, and the Layer 2 frame has DMAC as the gateway MAC of Spine-1. Then, Spine-1 routes the packet to VM3, where the ARP resolution is through the VXLAN configuration.
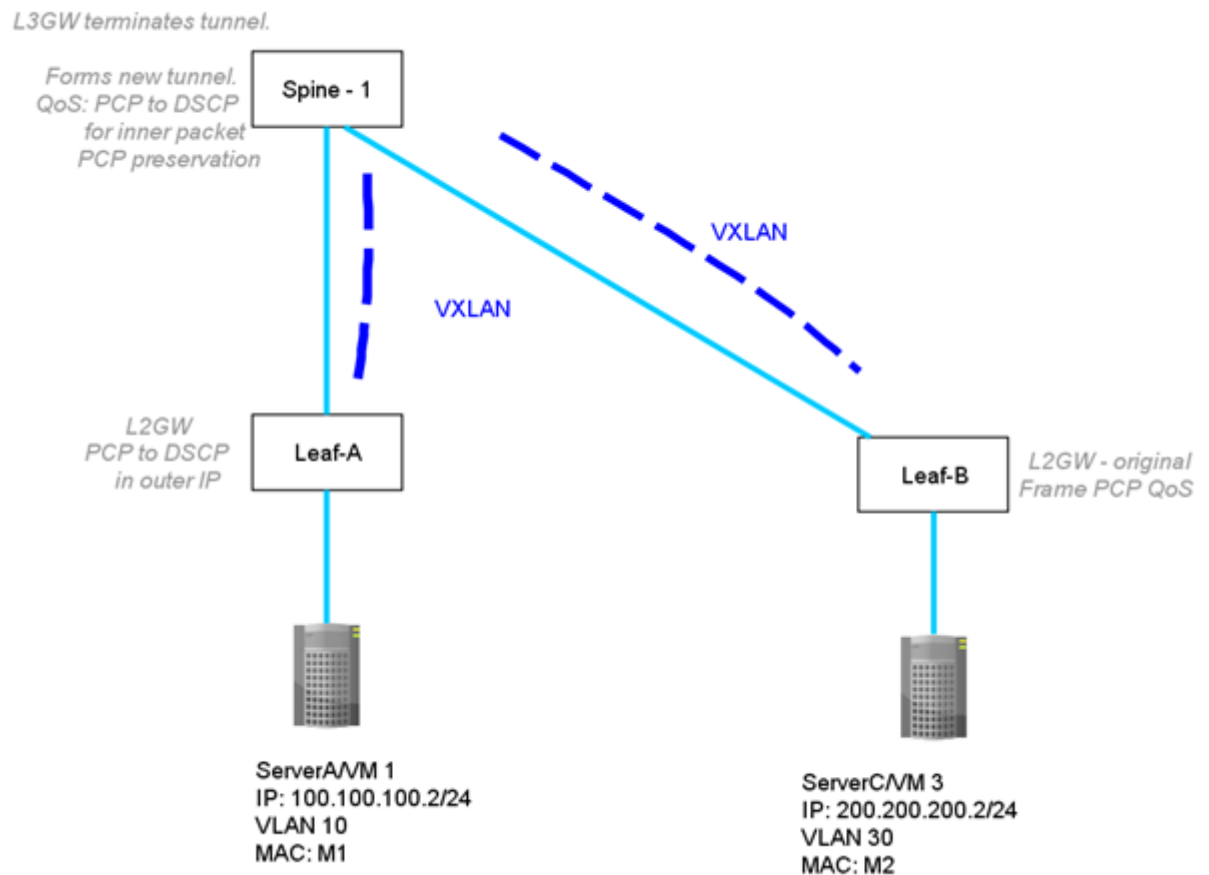
Figure 70: QoS for VXLAN Layer 2 and Layer 3 gateway interconnections

Configuration example

The **qos-ttl-mode uniform** command sets the QoS type mode to uniform, which is the default. The following example configures uniform QoS support on a VXLAN gateway.

```
device(config)# overlay-gateway gateway_L2L3
device(config-overlay-gw-gateway_L2L3)# type layer2-extension
device(config-overlay-gw-gateway_L2L3)# ip interface loopback 1
device(config-overlay-gw-gateway_L2L3)# qos-ttl-mode uniform
device(config-overlay-gw-gateway_L2L3)# map vni auto
device(config-overlay-gw-gateway_L2L3)# activate
```

QoS for VXLAN Layer 3 Gateways

SLX-OS VXLAN Layer 3 gateways can support QoS.

A VXLAN Layer 3 gateway allows inter-subnet communication through the VXLAN configuration. In the following figure, the leaf nodes act as the VXLAN Layer 3 gateway. Spine-1 and Spine-2 act as IP routing nodes that route the VXLAN packet out between Leaf-A and Leaf-B. Hosts on different subnets cannot learn MAC addresses of each other in a VXLAN network.

The leaf nodes resolve ARP for a tenant VM. The leaf nodes advertise the host routes to each other and tenant VMs with the next hop configured as VTEP. In the following figure, VM1 on a VXLAN network is sending a packet to VM3 on a VXLAN network. At Leaf-A, the DSCP of the tenant packet is propagated to the outer IP header. Leaf-A also adds a Layer 2 header corresponding to the VTEP gateway MAC, after the VXLAN header.

Optionally, the 802.1p VLAN and PCP marking of this inner Ethernet header can be set to a value corresponding to the received Layer 2 traffic class of the tenant packet. The Leaf-B terminating IP and Layer 2 headers ignore the QoS marking of transport network and proceed with marking the inner IP header.
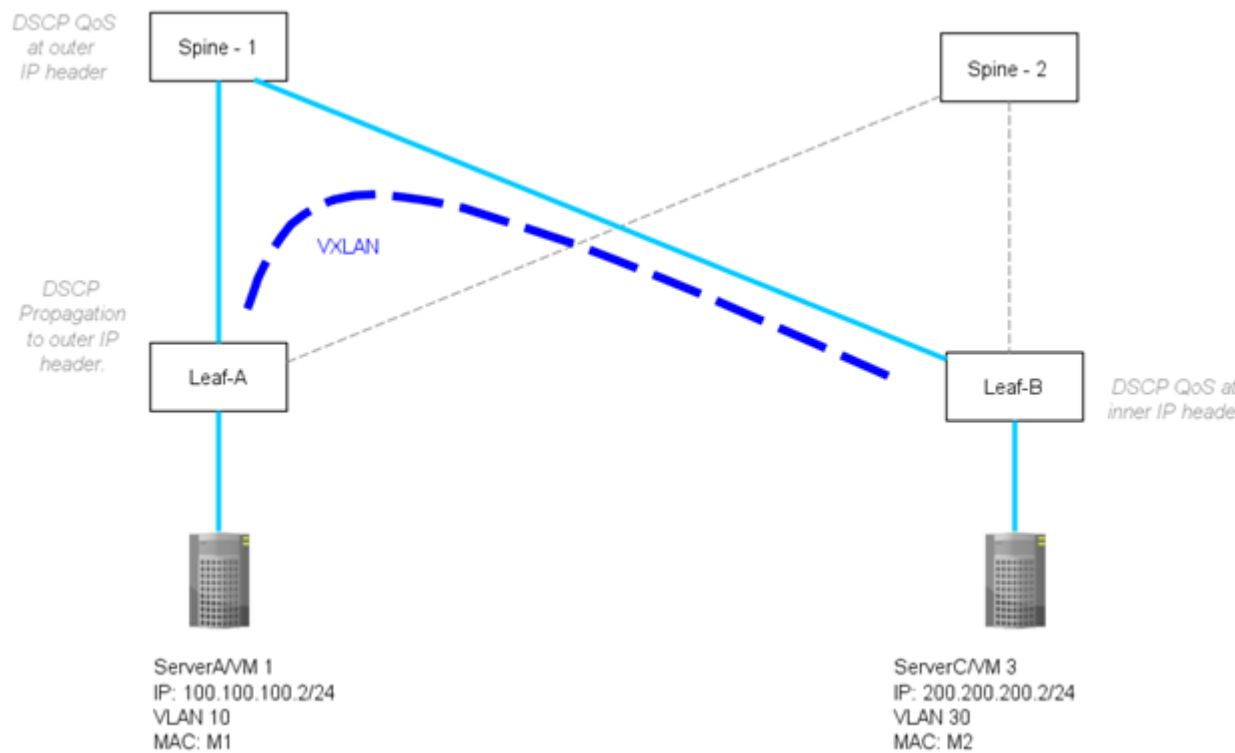

Figure 71: VXLAN L3 gateway support of intersubnet communication

Configuration example

A prerequisite for configuring QoS is to configure the TCAM profile that supports Layer 3 gateway. For more information, see Configure the TCAM Profile for Layer 3 Gateway on page 534.

The **qos-ttl-mode uniform** command sets the QoS type mode to uniform, which is the default. The following example configures uniform QoS support on a VXLAN gateway.

```
device(config)# overlay-gateway gateway_L3
device(config-overlay-gw-gateway_L3)# type layer2-extension
device(config-overlay-gw-gateway_L3)# ip interface loopback 1
device(config-overlay-gw-gateway_L3)# qos-ttl-mode uniform
device(config-overlay-gw-gateway_L3)# map vni auto
device(config-overlay-gw-gateway_L3)# activate
```