Junos® OS

Layer 3 VPNs User Guide for Routing Devices

Published
2024-12-19

Juniper Networks, Inc.
1133 Innovation Way
Sunnyvale, California 94089
USA
408-745-2000
www.juniper.net

Junos® OS Layer 3 VPNs User Guide for Routing Devices

The information in this document is current as of the date on the title page.

YEAR 2000 NOTICE

Juniper Networks hardware and software products are Year 2000 compliant. Junos OS has no known time-related limitations through the year 2038. However, the NTP application is known to have some difficulty in the year 2036.

END USER LICENSE AGREEMENT

The Juniper Networks product that is the subject of this technical documentation consists of (or is intended for use with) Juniper Networks software. Use of such software is subject to the terms and conditions of the End User License Agreement ("EULA") posted at https://support.juniper.net/support/eula/. By downloading, installing or using such software, you agree to the terms and conditions of that EULA.

Table of Contents

1 Layer 3 VPNs

2

Configuration Statements and Operational Commands

About This Guide

The Junos operating system (Junos OS) supports layer 3 VPN service which allows customers to have geographically dispersed private networks across service provider's networks. Use the topics on this page to configure VPN routing and forwarding instances to support Layer 3 VPNs.

RELATED DOCUMENTATION

Learn About Secure VPNs

Day One Poster: VPNs

FAQ: Layer 3 VPN on MX Series Routers

vDay One: Introduction to BGP Multicast VPNs

This Week: Deploying BGP Multicast VPNs

1

PART

Layer 3 VPNs

CHAPTER 1

Overview of Layer 3 VPNs

IN THIS CHAPTER

Overview

IN THIS SECTION

Types of VPNs

IN THIS SECTION

A virtual private network (VPN) consists of two topological areas: the provider’s network and the customer’s network. The customer’s network is commonly located at multiple physical sites and is also private (non-Internet). A customer site would typically consist of a group of routers or other networking equipment located at a single physical location. The provider’s network, which runs across the public Internet infrastructure, consists of routers that provide VPN services to a customer’s network as well as routers that provide other services. The provider’s network connects the various customer sites in what appears to the customer and the provider to be a private network.

To ensure that VPNs remain private and isolated from other VPNs and from the public Internet, the provider’s network maintains policies that keep routing information from different VPNs separate. A provider can service multiple VPNs as long as its policies keep routes from different VPNs separate. Similarly, a customer site can belong to multiple VPNs as long as it keeps routes from the different VPNs separate.

The Junos® Operating System (Junos OS) provides several types of VPNs; you can choose the best solution for your network environment. Each of the following VPNs has different capabilities and requires different types of configuration:

Layer 2 VPNs

Implementing a Layer 2 VPN on a router is similar to implementing a VPN using a Layer 2 technology such as ATM or Frame Relay. However, for a Layer 2 VPN on a router, traffic is forwarded to the router in Layer 2 format. It is carried by MPLS over the service provider’s network and then converted back to Layer 2 format at the receiving site. You can configure different Layer 2 formats at the sending and receiving sites. The security and privacy of an MPLS Layer 2 VPN are equal to those of an ATM or Frame Relay VPN.

On a Layer 2 VPN, routing occurs on the customer’s routers, typically on the CE router. The CE router connected to a service provider on a Layer 2 VPN must select the appropriate circuit on which to send traffic. The PE router receiving the traffic sends it across the service provider’s network to the PE router connected to the receiving site. The PE routers do not need to store or process the customer’s routes; they only need to be configured to send data to the appropriate tunnel.

For a Layer 2 VPN, customers need to configure their own routers to carry all Layer 3 traffic. The service provider needs to know only how much traffic the Layer 2 VPN needs to carry. The service provider’s routers carry traffic between the customer’s sites using Layer 2 VPN interfaces. The VPN topology is determined by policies configured on the PE routers.

Layer 3 VPNs

In a Layer 3 VPN, the routing occurs on the service provider's routers. Therefore, Layer 3 VPNs require more configuration on the part of the service provider, because the service provider's PE routers must store and process the customer's routes.

In the Junos OS, Layer 3 VPNs are based on RFC 4364, *BGP/MPLS IP Virtual Private Networks (VPNs)*. This RFC defines a mechanism by which service providers can use their IP backbones to provide Layer 3 VPN services to their customers. The sites that make up a Layer 3 VPN are connected over a provider's existing public Internet backbone.

VPNs based on RFC 4364 are also known as BGP/MPLS VPNs because BGP is used to distribute VPN routing information across the provider's backbone, and MPLS is used to forward VPN traffic across the backbone to remote VPN sites.

Customer networks, because they are private, can use either public addresses or private addresses, as defined in RFC 1918, *Address Allocation for Private Internets*. When customer networks that use private addresses connect to the public Internet infrastructure, the private addresses might overlap with the private addresses used by other network users. BGP/MPLS VPNs solve this problem by prefixing a VPN identifier to each address from a particular VPN site, thereby creating an address that is unique both within the VPN and within the public Internet. In addition, each VPN has its own VPN-specific routing table that contains the routing information for that VPN only.

VPLS

Virtual private LAN service (VPLS) allows you to connect geographically dispersed customer sites as if they were connected to the same LAN. In many ways, it works like a Layer 2 VPN. VPLS and Layer 2 VPNs use the same network topology and function similarly. A packet originating within a customer's network is sent first to a CE device. It is then sent to a PE router within the service provider's network. The packet traverses the service provider's network over an MPLS LSP. It arrives at the egress PE router, which then forwards the traffic to the CE device at the destination customer site.

The key difference in VPLS is that packets can traverse the service provider's network in a point-to-multipoint fashion, meaning that a packet originating from a CE device can be broadcast to PE routers in the VPLS. In contrast, a Layer 2 VPN forwards packets in a point-to-point fashion only. The destination of a packet received from a CE device by a PE router must be known for the Layer 2 VPN to function properly.

In a Layer 3 network only, you can configure virtual private LAN service (VPLS), to connect geographically dispersed Ethernet local area networks (LAN) sites to each other across an MPLS backbone. For ISP customers who implement VPLS, all sites appear to be in the same Ethernet LAN even though traffic travels across the service provider's network. VPLS is designed to carry Ethernet traffic across an MPLS-enabled service provider network. In certain ways, VPLS mimics the behavior of an Ethernet network. When a PE router configured with a VPLS routing instance receives a packet from a CE device, it first checks the appropriate routing table for the destination of the VPLS packet. If the router has the destination, it forwards it to the appropriate PE router. If it does not have the destination, it broadcasts the packet to all the other PE routers that are members of the same VPLS routing instance. The PE routers forward the packet to their CE devices. The CE device that is the intended recipient of the packet forwards it to its final destination. The other CE devices discard it.

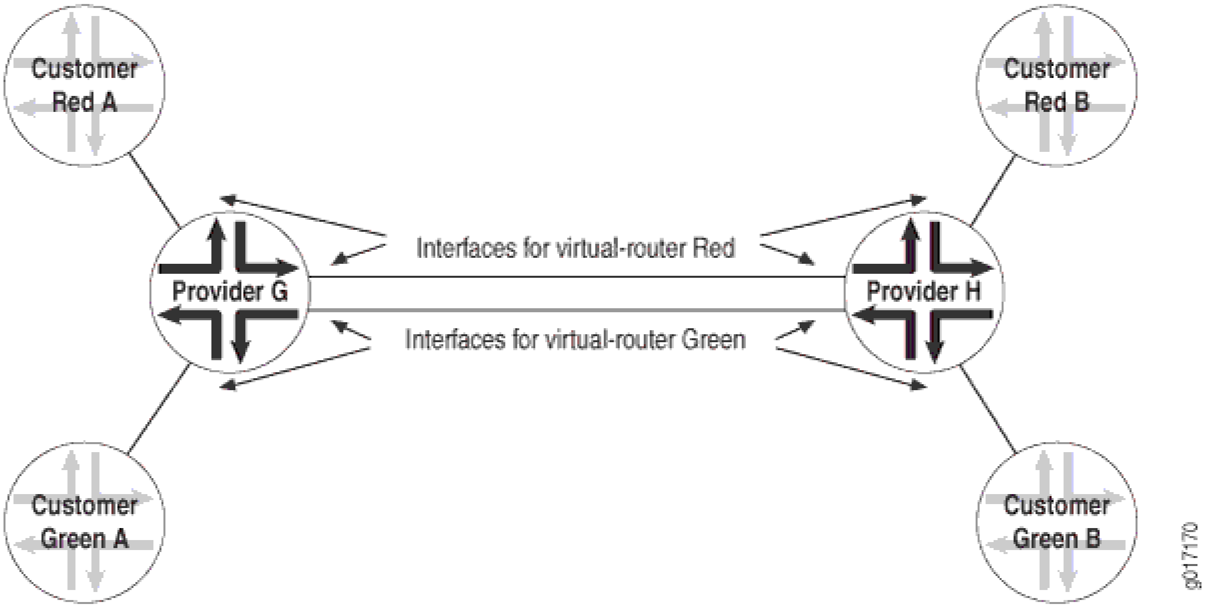
Virtual-Router Routing Instances

A virtual-router routing instance, like a VPN routing and forwarding (VRF) routing instance, maintains separate routing and forwarding tables for each instance. However, many configuration steps required for VRF routing instances are not required for virtual-router routing instances. Specifically, you do not need to configure a route distinguisher, a routing table policy (the `vrf-export`, `vrf-import`, and `route-distinguisher` statements), or MPLS between the P routers.

However, you need to configure separate logical interfaces between each of the service provider routers participating in a virtual-router routing instance. You also need to configure separate logical interfaces between the service provider routers and the customer routers participating in each routing instance. Each virtual-router instance requires its own unique set of logical interfaces to all participating routers.

Figure 1 on page 5 shows how this works. The service provider routers G and H are configured for virtual-router routing instances Red and Green. Each service provider router is directly connected to two local customer routers, one in each routing instance. The service provider routers are also connected to each other over the service provider network. These routers need four logical interfaces: a *logical interface* to each of the locally connected customer routers and a logical interface to carry traffic between the two service provider routers for each virtual-router instance.

Figure 1: Logical Interface per Router in a Virtual-Router Routing Instance



Layer 3 VPNs do not have this configuration requirement. If you configure several Layer 3 VPN routing instances on a PE router, all the instances can use the same logical interface to reach another PE router. This is possible because Layer 3 VPNs use MPLS (VPN) labels that differentiate traffic going to and from various routing instances. Without MPLS and VPN labels, as in a virtual-router routing instance, you need separate logical interfaces to separate traffic from different instances.

One method of providing this logical interface between the service provider routers is by configuring tunnels between them. You can configure IP Security (IPsec), generic routing encapsulation (GRE), or IP-IP tunnels between the service provider routers, terminating the tunnels at the virtual-router instance.

VPNs and Logical Systems

You can partition a single physical router into multiple logical systems that perform independent routing tasks. Because logical systems perform a subset of the tasks once handled by the physical router, logical systems offer an effective way to maximize the use of a single routing platform.

Logical systems perform a subset of the actions of a physical router and have their own unique routing tables, interfaces, policies, and routing instances. A set of logical systems within a single router can handle the functions previously performed by several small routers.

Logical systems support Layer 2 VPNs, Layer 3 VPNs, VPLS, and Layer 2 circuits. For more information about logical systems, see the Logical Systems User Guide for Routers and Switches.

Starting in Junos OS release 17.4R1, Ethernet VPN (EVPN) support has also been extended to logical systems running on MX devices. The same EVPN options and performance are available, and can be configured under the `[edit logical-systems` *`logical-system-name`* `routing-instances` *`routing-instance-name`* `protocols evpn]` hierarchy.

Understanding Layer 3 VPNs

IN THIS SECTION

Virtual private networks (VPNs) are private networks that use a public network to connect two or more remote sites. Instead of dedicated connections between networks, VPNs use virtual connections routed (tunneled) through public networks that are typically service provider networks.

Layer 3 VPN operates at the Layer 3 level of the OSI model, the Network layer. A Layer 3 VPN is composed of a set of customer sites that are connected over a service provider's existing public Internet backbone. A peer-to-peer model is used to connect to the customer sites, where the service providers learn the customer routes on peering with the customers. The common routing information is shared across the provider's backbone using multiprotocol BGP, and the VPN traffic is forwarded to the customer sites using MPLS.

Junos OS supports Layer 3 VPNs based on RFC 4364. The RFC describes VPNs using MPLS tunnels for connectivity, BGP to distribute reachability information, and an IP backbone for transport. Service providers use their IP backbones to link a set of customer sites belonging to the same VPN.

Components of a Layer 3 VPN

There are three primary types of MPLS VPNs: Layer 2 VPNs, Layer 2 circuits, and Layer 3 VPNs. All types of MPLS VPNs share certain components:

- CE devices—Customer Edge (CE) devices at the customer premises that connect to the provider’s network. Some models call these Customer Premises Equipment (CPE) devices.

- Customer network—Customer sites with CE devices that belong to the VPN.

- Provider network—The service provider backbone network running the MPLS backbone.

- P devices—Provider (P) devices within the core of the provider’s network. Provider devices are not connected to any device at a customer site and are part of the tunnel between pairs of PE devices. Provider devices support label-switched path (LSP) functionality as part of the tunnel support, but do not support VPN functionality.

- PE devices—Provider Edge (PE) devices within a service provider core network that connect directly to a CE device at the customer's site.

- MP-BGP— PE devices use MP-BGP to distribute customer routes to the proper PE devices across the MPLS backbone.

Layer 3 VPN Terminology

VPNs use a distinct terminology to identify components of the network:

- IP routing table (also called the global routing table)—This table contains service provider routes not included in a VRF. Provider devices need this table to be able to reach each other, while the VRF table is needed to reach all customer devices on a particular VPN. For example, a PE router with Interface A to a CE router and Interface B to a backbone P router places the Interface A addresses in the VRF and the Interface B addresses in the global IP routing table.

- Route Distinguisher—A 64-bit value prepended to an IP address. This unique tag helps identify the different customers’ routes as packets flow across the same service provider tunnel.

Because a typical transit network is configured to handle more than one VPN, the provider routers are likely to have multiple VRF instances configured. As a result, depending on the origin of the traffic and any filtering rules applied to the traffic, the BGP routing tables can contain multiple routes for a particular destination address. Because BGP requires that exactly one BGP route per destination to be imported into the forwarding table, BGP must have a way to distinguish between potentially identical network layer reachability information (NLRI) messages received from different VPNs.

A route distinguisher is a locally unique number that identifies all route information for a particular VPN. Unique numeric identifiers allow BGP to distinguish between routes that are otherwise identical.

Each routing instance that you configure on a PE router must have a unique route distinguisher. There are two possible formats:

  - **as-number:number**—Where, *as-number* is an autonomous system (AS) number (a 2–byte value) in the range 1 through 65,535, and *number* is any 4–byte value. We recommend that you use an Internet Assigned Numbers Authority (IANA)-assigned, nonprivate AS number, preferably the ISP or the customer AS number.

  - **ip-address:number**—Where, *ip-address* is an IP address (a 4–byte value), and *number* is any 2–byte value. The IP address can be any globally unique unicast address. We recommend that you use the address that you configure in the router-id statement, which is a public IP address in your assigned prefix range.

- Route Target (RT)—A 64-bit value used to identify the final egress PE device for customer routes in a particular VRF to enable complex sharing of routes. The route target defines which route is part of a VPN. A unique route target helps distinguish between different VPN services on the same router. Each VPN also has a policy that defines how routes are imported into the VRF table on the router. A Layer 2 VPN is configured with import and export policies. A Layer 3 VPN uses a unique route target to distinguish between VPN routes. For example, the RT enables the sharing of routes in a shared service network to multiple customers. Each VPN route can have one or more RTs. A PE device handles RTs as extended BGP community values and uses the RTs to install customer routes.

- VPN-IPv4 routes—A route consisting of a 96-bit sequence composed of a 64-bit RD tag prepended to a 32-bit IPv4 address. The PE devices export the VPN-IPv4 routes in IBGP sessions to the other provider devices.These routes are exchanged across the MPLS backbone using iBGP. When the outbound PE device receives the route, it strips off the route distinguisher and advertises the route to the connected CE devices, typically through standard BGP IPv4 route advertisements.

- VRF—The virtual routing and forwarding (VRF) table distinguishes the routes for different customers, as well as customer routes from provider routes on the PE device. These routes can include overlapping private network address spaces, customer-specific public routes, and provider routes on a PE device useful to the customer.

  A VRF instance consists of one or more routing tables, a derived forwarding table, the interfaces that use the forwarding table, and the policies and routing protocols that determine what goes into the forwarding table. Because each instance is configured for a particular VPN, each VPN has separate tables, rules, and policies that control its operation.

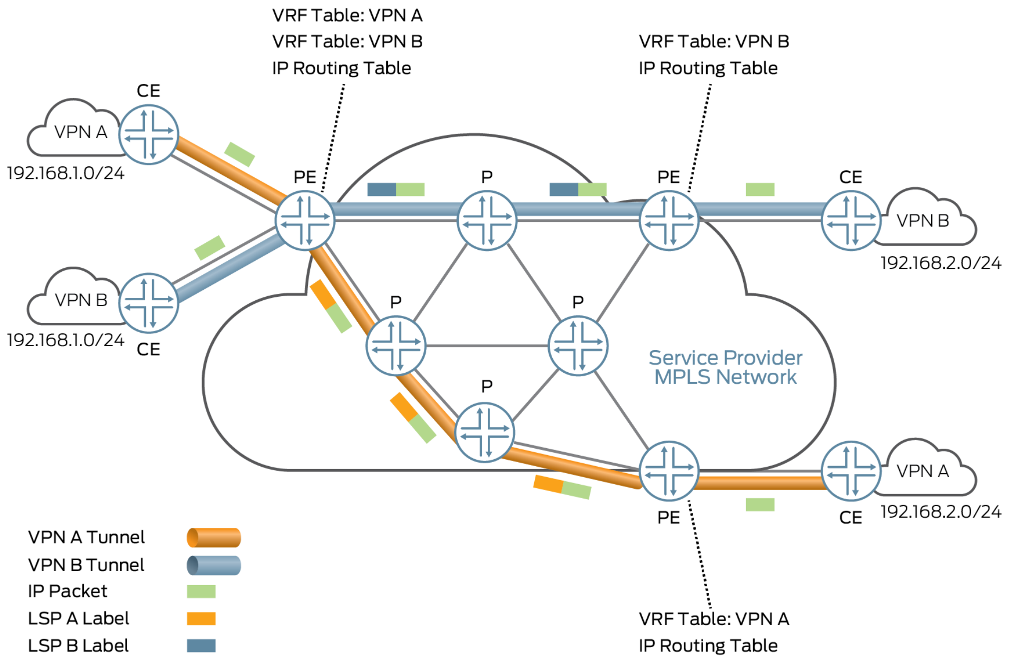
A separate VRF table is created for each VPN that has a connection to a CE router. The VRF table is populated with routes received from directly connected CE sites associated with the VRF instance, and with routes received from other PE routers in the same VPN.

Layer 3 VPN Architecture

A Layer 3 VPN links customer-edge routers (CE routers) to routers on the edge of the service provider network (PE routers). A Layer 3 VPN uses a peer routing model between local PE and CE routers that directly connect. That is, without needing multiple hops on the provider backbone to connect PE and CE router pairs. The PE routers distribute routing information to all CE routers belonging to the same VPN, based on the BGP route distinguisher, locally and across the provider network. Each VPN has its own routing table for that VPN, coordinated with the routing tables in the CE and PE peer routers. The CE and PE routers have different VRF tables. Each CE router has only a single VRF table because the other VPNs are invisible to the CE. A PE router can connect to more than one CE router, so the PE router has a general IP routing table and VRF table for each attached CE with a VPN.

Figure 2 on page 9 shows the general architecture of a Layer 3 VPN.

Figure 2: General Layer 3 VPN Architecture.



The PE router knows which VRF table to use for packets arriving from remote VPN sites because every VRF table has one or more extended community attributes associated with hit. The community attributes identify the route as belonging to a specific collection of routers. The route target community attribute identifies a collection of sites (more accurately, the collection of their VRF tables) to which a PE router distributes routes. The PE router uses the route target to import the correct remote VPN routes into its VRF tables.

The import and export of VPN routes between VPN sites is not automatic. This process is controlled by BPG routing polices. The routing policies establish the rules for exchanging routing information across the service provider's MPLS network and must be configured correctly and maintained when the network topology changes.

The PE router classifies IPv4 routes announced by a peer CE router and received by the PE router as VPN-IPv4 routes. When an ingress PE router receives routes advertised from a directly connected peer CE router, the ingress PE router checks the received route against the VRF export policy for that VPN. That is, the ingress PE router decides which remote PE routers need to know about the advertised routes. This is a two-step process:

- If the established export policy accepts the route, the PE router converts the information to VPN-IPv4 format by adding the route distinguisher to the IPv4 address. The PE router then announces the VPN-IPv4 route to the remote PE routers. The configured export target policy of the VRF table determines the value of the attached route target. IBGP sessions distribute the VNP-IPv4 routes across the service provider's core network.

- If the established export policy does not accept the route, the PE router does not export the route to other PE routers, but the PE router uses the route locally. This happens, for example, when two CE routers in the same VPN connect directly to the same PE router so general traffic can flow from one CE site to another.

When an egress PE router on the other side of the service provider network receives a route, the egress PE router checks the route against the IBGP import policy in place between the PE routers. If the egress PE router accepts the route, then the egress PE router adds the route to the bgp.l3vpn.0 routing table. The router also checks the route against the VRF import policy for the VPN. If the route is accepted, the egress PE router removes the route distinguisher and places the route into the correct VRF table. The VRF tables use the routing-instance-name.inet.0 naming convention, so "VPN A" usually configures the table as vpna.inet.0).

Supported Layer 3 VPN Standards

Junos OS substantially supports the following RFCs, which define standards for Layer 3 virtual private networks (VPNs).

- RFC 2283, *Multiprotocol Extensions for BGP-4*

- RFC 2685, *Virtual Private Networks Identifier*

- RFC 2858, *Multiprotocol Extensions for BGP-4*
- RFC 4364, *BGP/MPLS IP Virtual Private Networks (VPNs)*
- RFC 4379, *Detecting Multi-Protocol Label Switched (MPLS) Data Plane Failures*

  The traceroute functionality is supported only on transit routers.
- RFC 4576, *Using a Link State Advertisement (LSA) Options Bit to Prevent Looping in BGP/MPLS IP Virtual Private Networks (VPNs)*
- RFC 4577, *OSPF as the Provider/Customer Edge Protocol for BGP/MPLS IP Virtual Private Networks (VPNs)*
- RFC 4659, *BGP-MPLS IP Virtual Private Network (VPN) Extension for IPv6 VPN*
- RFC 4684, *Constrained Route Distribution for Border Gateway Protocol/MultiProtocol Label Switching (BGP/MPLS) Internet Protocol (IP) Virtual Private Networks (VPNs)*

The following RFCs do not define a standard, but provide information about technology related to Layer 3 VPNs. The IETF classifies them as a “Best Current Practice” or “Informational.”

- RFC 1918, *Address Allocation for Private Internets*
- RFC 2917, *A Core MPLS IP VPN Architecture*

SEE ALSO

Supported Carrier-of-Carriers and Interprovider VPN Standards | **446**

Supported VPWS Standards

Supported Layer 2 VPN Standards

Supported Multicast VPN Standards | **583**

Supported VPLS Standards

Supported MPLS Standards

Supported Standards for BGP

Accessing Standards Documents on the Internet

Understanding Layer 3 VPN Forwarding Through the Core

The PE routers in the provider’s core network are the only routers that are configured to support VPNs and hence are the only routers to have information about the VPNs. From the point of view of VPN functionality, the provider (P) routers in the core—those P routers that are not directly connected to CE routers—are merely routers along the tunnel between the ingress and egress PE routers.

The tunnels can be either LDP or MPLS. Any P routers along the tunnel must support the protocol used for the tunnel, either LDP or MPLS.

When PE-router-to-PE router forwarding is tunneled over MPLS label-switched paths (LSPs), the MPLS packets have a two-level label stack (see Figure 3 on page 12):

- Outer label—Label assigned to the address of the BGP next hop by the IGP next hop
- Inner label—Label that the BGP next hop assigned for the packet's destination address

Figure 3: Using MPLS LSPs to Tunnel Between PE Routers

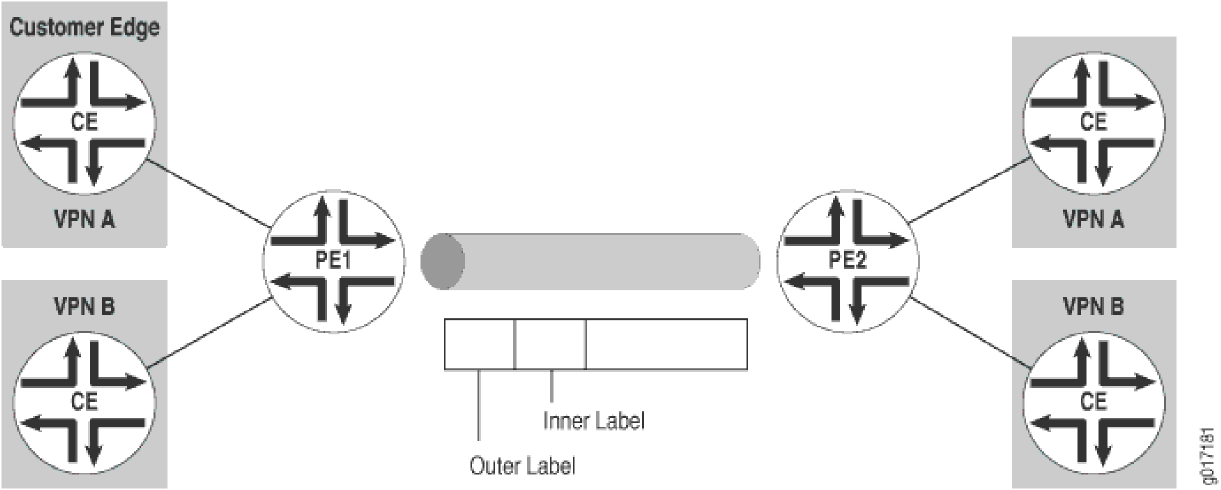
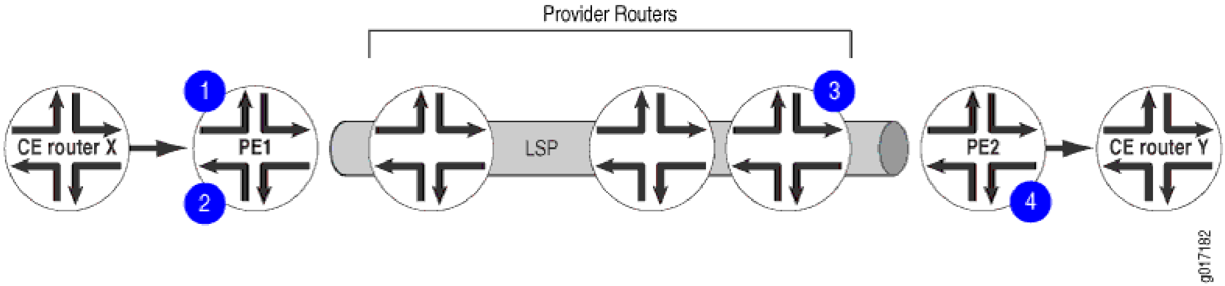


Figure 4 on page 13 illustrates how the labels are assigned and removed:

1. When CE Router X forwards a packet to Router PE1 with a destination of CE Router Y, the PE route identifies the BGP next hop to Router Y and assigns a label that corresponds to the BGP next hop and identifies the destination CE router. This label is the inner label.
2. Router PE1 then identifies the IGP route to the BGP next hop and assigns a second label that corresponds to the LSP of the BGP next hop. This label is the outer label.
3. The inner label remains the same as the packet traverses the LSP tunnel. The outer label is swapped at each hop along the LSP and is then popped by the penultimate hop router (the third P router).
4. Router PE2 pops the inner label from the route and forwards the packet to Router Y.

Figure 4: Label Stack



Understanding Layer 3 VPN Attributes

Route distribution within a VPN is controlled through BGP extended community attributes. RFC 4364 defines the following three attributes used by VPNs:

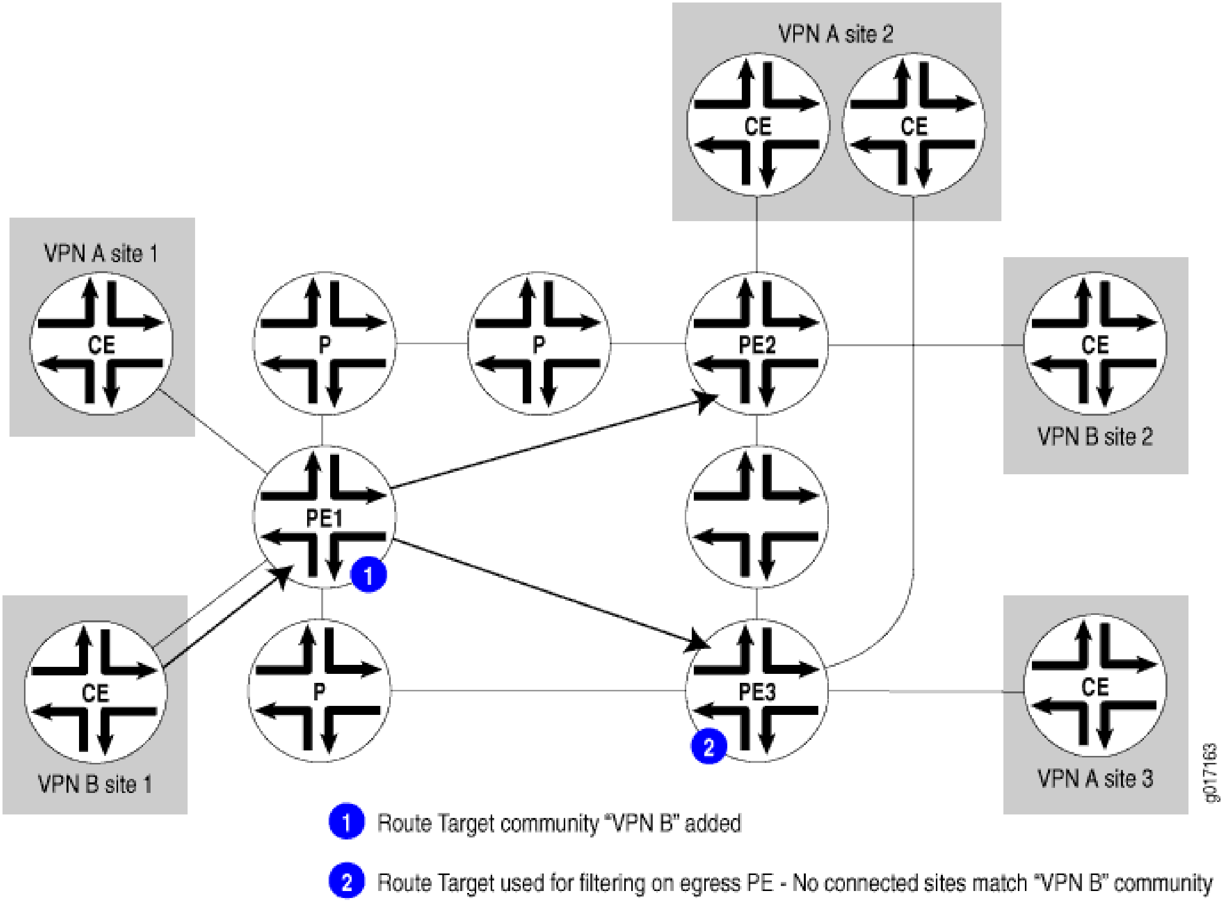
- Target VPN—Identifies a set of sites within a VPN to which a provider edge (PE) router distributes routes. This attribute is also called the *route target*. The route target is used by the egress PE router to determine whether a received route is destined for a VPN that the router services.

  Figure 5 on page 14 illustrates the function of the route target. PE Router PE1 adds the route target "VPN B" to routes received from the customer edge (CE) router at Site 1 in VPN B. When it receives the route, the egress router PE2 examines the route target, determines that the route is for a VPN that it services, and accepts the route. When the egress router PE3 receives the same route, it does not accept the route because it does not service any CE routers in VPN B.

- VPN of origin—Identifies a set of sites and the corresponding route as having come from one of the sites in that set.

- Site of origin—Uniquely identifies the set of routes that a PE router learned from a particular site. This attribute ensures that a route learned from a particular site through a particular PE-CE connection is not distributed back to the site through a different PE-CE connection. It is particularly useful if you are using BGP as the routing protocol between the PE and CE routers and if different sites in the VPN have been assigned the same autonomous system (AS) numbers.

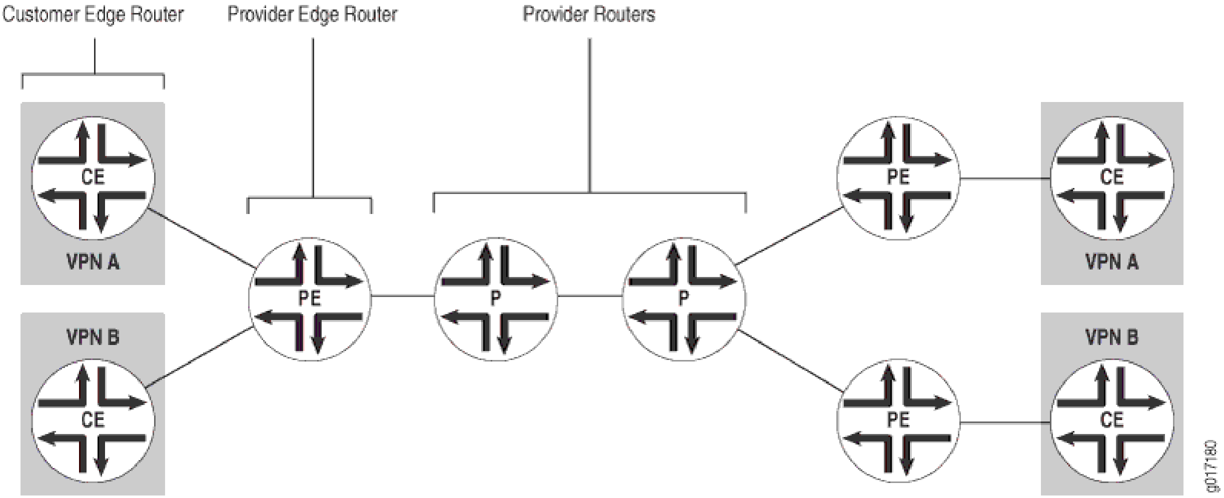
Figure 5: VPN Attributes and Route Distribution



Routers in a VPN

Figure 6 on page 15 illustrates how VPN functionality is provided by the provider edge (PE) routers; the provider and customer edge (CE) routers have no special configuration requirements for VPNs.

Figure 6: Routers in a VPN



Introduction to Configuring Layer 3 VPNs

To configure Layer 3 virtual private network (VPN) functionality, you must enable VPN support on the provider edge (PE) router. You must also configure any provider (P) routers that service the VPN, and you must configure the customer edge (CE) routers so that their routes are distributed into the VPN.

To configure Layer 3 VPNs, you include the following statements:

```
description text;
instance-type vrf;
interface interface-name;
protocols {
    bgp {
        group group-name {
            peer-as as-number;
            neighbor ip-address;
        }
        multihop ttl-value;
    }
    (ospf | ospf3) {
        area area {
            interface interface-name;
        }
        domain-id domain-id;
        domain-vpn-tag number;
        sham-link {
            local address;
```

```
        }
        sham-link-remote address <metric number>;
    }
    rip {
        rip-configuration;
    }
}
route-distinguisher (as-number:id | ip-address:id);
router-id address;
routing-options {
    autonomous-system autonomous-system {
        independent-domain;
        loops number;
    }
    forwarding-table {
        export [ policy-names ];
    }
    interface-routes {
        rib-group group-name;
    }
    martians {
        destination-prefix match-type <allow>;
    }
    maximum-paths {
        path-limit;
        log-interval interval;
        log-only;
        threshold percentage;
    }
    maximum-prefixes {
        prefix-limit;
        log-interval interval;
        log-only;
        threshold percentage;
    }
    multipath {
        vpn-unequal-cost;
    }
    options {
        syslog (level level | upto level);
    }
    rib routing-table-name {
        martians {
```

```
            destination-prefix match-type <allow>;
        }
        multipath {
            vpn-unequal-cost;
        }
        static {
            defaults {
                static-options;
            }
            route destination-prefix {
                next-hop [next-hops];
                static-options;
            }
        }
    }
}
static {
    defaults {
        static-options;
    }
    route destination-prefix {
        policy [ policy-names ];
        static-options;
    }
}
vrf-advertise-selective {
    family {
        inet-mvpn;
        inet6-mvpn;
    }
}
vrf-export [ policy-names ];
vrf-import [ policy-names ];
vrf-target (community | export community-name | import community-name);
vrf-table-label;
```

You can include these statements at the following hierarchy levels:

- `[edit routing-instances` *`routing-instance-name`*`]`
- `[edit logical-systems` *`logical-system-name`* `routing-instances` *`routing-instance-name`*`]`

NOTE: The `[edit logical-systems]` hierarchy level is not applicable in ACX Series routers.

The `sham-link`, `sham-link-remote`, and `vrf-advertise-selective` statements are not applicable in ACX Series routers.

For Layer 3 VPNs, only some of the statements in the `[edit routing-instances]` hierarchy are valid. For the full hierarchy, see Junos OS Routing Protocols Library.

In addition to these statements, you must enable a signaling protocol, IBGP sessions between the PE routers, and an interior gateway protocol (IGP) on the PE and P routers.

By default, Layer 3 VPNs are disabled.

Many of the configuration procedures for Layer 3 VPNs are common to all types of VPNs.

Change History Table

Feature support is determined by the platform and release you are using. Use Feature Explorer to determine if a feature is supported on your platform.

Release	Description
17.4	Starting in Junos OS release 17.4R1, Ethernet VPN (EVPN) support has also been extended to logical systems running on MX devices. The same EVPN options and performance are available, and can be configured under the `[edit logical-systems` *`logical-system-name`* `routing-instances` *`routing-instance-name`* `protocols evpn]` hierarchy.

CHAPTER 2

Routing in Layer 3 VPNs

IN THIS CHAPTER

Routing Instances in Layer 3 VPNs

IN THIS SECTION

This topic discusses configuring routing instances in Layer 3 VPNs

Routing Instances in Layer 3 VPNs

A routing instance is a collection of routing tables, interfaces, and routing protocol parameters. The set of interfaces belongs to the routing tables, and the routing protocol parameters control the information in the routing tables. Each routing instance has a unique name and a corresponding IP unicast table.

To implement Layer 3 VPNs in the JUNOS Software, you configure one routing instance for each VPN. You configure the routing instances on PE routers only. Each VPN routing instance consists of the following components:

- VRF table—On each PE router, you configure one VRF table for each VPN.
- Set of interfaces that use the VRF table—The *logical interface* to each directly connected CE router must be associated with a VRF table. You can associate more than one interface with the same VRF table if more than one CE router in a VPN is directly connected to the PE router.
- Policy rules—These control the import of routes into and the export of routes from the VRF table.
- One or more routing protocols that install routes from CE routers into the VRF table—You can use the BGP, OSPF, and RIP routing protocols, and you can use static routes.

Configuring Logical Units on the Loopback Interface for Routing Instances in Layer 3 VPNs

For Layer 3 VPNs (VRF routing instances), you can configure a logical unit on the loopback interface into each VRF routing instance that you have configured on the router. Associating a VRF routing instance with a logical unit on the loopback interface allows you to easily identify the VRF routing instance.

Doing this is useful for troubleshooting:

- It allows you to ping a remote CE router from a local PE router in a Layer 3 VPN. For more information, see "Example: Troubleshooting Layer 3 VPNs" on page 1231.
- It ensures that a path maximum transmission unit (MTU) check on traffic originating on a VRF or virtual-router routing instance functions properly. For more information, see "Configuring Path MTU Checks for VPN Routing Instances" on page 30.

You can also configure a firewall filter for the logical unit on the loopback interface; this configuration allows you to filter traffic for the VRF routing instance associated with it.

The following describes how firewall filters affect the VRF routing instance depending on whether they are configured on the default loopback interface, the VRF routing instance, or some combination of the two. The “default loopback interface” refers to `lo0.0` (associated with the default routing table), and the “VRF loopback interface” refers to `lo0.`*n*, which is configured in the VRF routing instance.

- If you configure Filter A on the default loopback interface and Filter B on the VRF loopback interface, the VRF routing instance uses Filter B.
- If you configure Filter A on the default loopback interface but do not configure a filter on the VRF loopback interface, the VRF routing instance does not use a filter.
- If you configure Filter A on the default loopback interface but do not configure a VRF loopback interface, the VRF routing instance uses Filter A. For MX80 devices, the behavior is slightly different: If you configure filters on the default loopback interface but do not configure a VRF loopback interface, the VRF routing instance uses only the input filters assigned to the default loopback (it does not use output filters from the default loopback).

For some ACX Series Universal Metro Routers (ACX1000, ACX2000, ACX4000, and ACX5000), the default loopback filter must be in the same routing, or virtual routing and forwarding (VRF), instance as the ingress traffic it filters. That is, on these devices, the default loopback filter cannot be used for traffic traversing an interface that belongs to a different routing instance.

To configure a logical unit on the loopback interface, include the `unit` statement:

```
unit number {
    family inet {
        address address;
    }
}
```

You can include this statement at the following hierarchy levels:

- `[edit interfaces lo0]`
- `[edit logical-systems logical-system-name interfaces lo0]`

To associate a firewall filter with the logical unit on the loopback interface, include the `filter` statement:

```
filter {
    input filter-name;
}
```

You can include this statement at the following hierarchy levels:

- `[edit interfaces lo0 unit unit-number family inet]`
- `[edit logical-systems logical-system-name interfaces lo0 unit unit-number family inet]`

To include the `lo0.n` interface (where *n* specifies the logical unit) in the configuration for the VRF routing instance, include the following statement:

```
interface lo0.n;
```

You can include this statement at the following hierarchy levels:

- `[edit routing-instances routing-instance-name]`
- `[edit logical-systems logical-system-name routing-instances routing-instance-name]`

Configuring Routing Instances on PE Routers in VPNs

IN THIS SECTION

You need to configure a routing instance for each VPN on each of the PE routers participating in the VPN. The configuration procedures outlined in this section are applicable to Layer 2 VPNs, Layer 3 VPNs, and VPLS. The configuration procedures specific to each type of VPN are described in the corresponding sections in the other configuration chapters.

To configure routing instances for VPNs, include the following statements:

```
description text;
instance-type type;
interface interface-name;
route-distinguisher (as-number:number | ip-address:number);
vrf-import [ policy-names ];
vrf-export [ policy-names ];
vrf-target {
    export community-name;
```

```
    import community-name;
}
```

You can include these statements at the following hierarchy levels:

- `[edit routing-instances routing-instance-name]`
- `[edit logical-systems logical-system-name routing-instances routing-instance-name]`

To configure VPN routing instances, you perform the steps in the following sections:

Configuring the Routing Instance Name for a VPN

The name of the routing instance for a VPN can be a maximum of 128 characters and can contain letters, numbers, and hyphens. In Junos OS Release 9.0 and later, you can no longer specify `default` as the actual routing-instance name. You also cannot use any special characters (! @ # $ % ^ & * , +< > : ;) within the name of a routing instance.

NOTE: In Junos OS Release 9.6 and later, you can include a slash (/) in a routing instance name only if a logical system is not configured. That is, you cannot include the slash character in a routing instance name if a logical system other than the default is explicitly configured.

Specify the routing-instance name with the `routing-instance` statement:

```
routing-instance routing-instance-name {...}
```

You can include this statement at the following hierarchy levels:

- `[edit]`
- `[edit logical-systems logical-system-name]`

Configuring the Description

To provide a text description for the routing instance, include the `description` statement. If the text includes one or more spaces, enclose them in quotation marks (" "). Any descriptive text you include is displayed in the output of the `show route instance detail` command and has no effect on the operation of the routing instance.

To configure a text description, include the `description` statement:

```
description text;
```

You can include this statement at the following hierarchy levels:

- `[edit routing-instances routing-instance-name]`
- `[edit logical-systems logical-system-name routing-instances routing-instance-name]`

Configuring the Instance Type

The instance type you configure varies depending on whether you are configuring Layer 2 VPNs, Layer 3 VPNs, VPLS, or virtual routers. Specify the instance type by including the `instance-type` statement:

- To enable Layer 2 VPN routing on a PE router, include the `instance-type` statement and specify the value `l2vpn`:

  ```
  instance-type l2vpn;
  ```

- To enable VPLS routing on a PE router, include the `instance-type` statement and specify the value `vpls`:

  ```
  instance-type vpls;
  ```

- Layer 3 VPNs require that each PE router have a VPN routing and forwarding (VRF) table for distributing routes within the VPN. To create the VRF table on the PE router, include the `instance-type` statement and specify the value `vrf`:

  ```
  instance-type vrf;
  ```

  NOTE: Routing Engine based sampling is not supported on VRF routing instances.

- To enable the virtual-router routing instance, include the `instance-type` statement and specify the value `virtual-router`:

  ```
  instance-type virtual-router;
  ```

You can include this statement at the following hierarchy levels:

- `[edit routing-instances *routing-instance-name*]`
- `[edit logical-systems *logical-system-name* routing-instances *routing-instance-name*]`

Configuring Interfaces for VPN Routing

On each PE router, you must configure an interface over which the VPN traffic travels between the PE and CE routers.

The sections that follow describe how to configure interfaces for VPNs:

General Configuration for VPN Routing

The configuration described in this section applies to all types of VPNs. For Layer 3 VPNs and carrier-of-carriers VPNs, complete the configuration described in this section before proceeding to the interface configuration sections specific to those topics.

To configure interfaces for VPN routing, include the `interface` statement:

```
interface interface-name;
```

You can include this statement at the following hierarchy levels:

- `[edit routing-instances *routing-instance-name*]`
- `[edit logical-systems *logical-system-name* routing-instances *routing-instance-name*]`

Specify both the physical and logical portions of the interface name, in the following format:

```
physical.logical
```

For example, in `at-1/2/1.2`, `at-1/2/1` is the physical portion of the interface name and `2` is the logical portion. If you do not specify the logical portion of the interface name, the value `0` is set by default.

A logical interface can be associated with only one routing instance. If you enable a routing protocol on all instances by specifying `interfaces all` when configuring the master instance of the protocol at the `[edit protocols]` hierarchy level, and if you configure a specific interface for VPN routing at the `[edit routing-instances` *`routing-instance-name`*`]` hierarchy level or at the `[edit logical-systems` *`logical-system-name`* `routing-instances` *`routing-instance-name`*`]` hierarchy level, the latter interface statement takes precedence and the interface is used exclusively for the VPN.

If you explicitly configure the same interface name at the `[edit protocols]` hierarchy level and at either the `[edit routing-instances` *`routing-instance-name`*`]` or `[edit logical-systems` *`logical-system-name`* `routing-instances` *`routing-instance-name`*`]` hierarchy levels, an attempt to commit the configuration fails.

Configuring Interfaces for Layer 3 VPNs

When you configure the Layer 3 VPN interfaces at the `[edit interfaces]` hierarchy level, you must also configure `family inet` when configuring the logical interface:

```
[edit interfaces]
interface-name {
    unit logical-unit-number {
        family inet;
    }
}
```

Configuring Interfaces for Carrier-of-Carriers VPNs

When you configure carrier-of-carriers VPNs, you need to configure the `family mpls` statement in addition to the `family inet` statement for the interfaces between the PE and CE routers. For carrier-of-carriers VPNs, configure the logical interface as follows:

```
[edit interfaces]
interface-name {
    unit logical-unit-number {
        family inet;
        family mpls;
    }
}
```

If you configure `family mpls` on the logical interface and then configure this interface for a non-carrier-of-carriers routing instance, the `family mpls` statement is automatically removed from the configuration for the logical interface, since it is not needed.

Configuring Unicast RPF on VPN Interfaces

For VPN interfaces that carry IP version 4 or version 6 (IPv4 or IPv6) traffic, you can reduce the impact of denial-of-service (DoS) attacks by configuring unicast reverse path forwarding (RPF). Unicast RPF helps determine the source of attacks and rejects packets from unexpected source addresses on interfaces where unicast RPF is enabled.

You can configure unicast RPF on a VPN interface by enabling unicast RPF on the interface and including the `interface` statement at the `[edit routing-instances routing-instance-name]` hierarchy level.

You cannot configure unicast RPF on the core-facing interfaces. You can only configure unicast RPF on the CE router-to-PE router interfaces on the PE router. However, for virtual-router routing instances, unicast RPF is supported on all interfaces you specify in the routing instance.

For information about how to configure unicast RPF on VPN interfaces, see Understanding Unicast RPF (Routers).

Configuring the Route Distinguisher

Each routing instance that you configure on a PE router must have a unique route distinguisher associated with it. VPN routing instances need a route distinguisher to help BGP to distinguish between potentially identical network layer reachability information (NLRI) messages received from different VPNs. If you configure different VPN routing instances with the same route distinguisher, the commit fails.

For Layer 2 VPNs and VPLS, if you have configured the `l2vpn-use-bgp-rules` statement, you must configure a unique route distinguisher for each PE router participating in a specific routing instance.

For other types of VPNs, we recommend that you use a unique route distinguisher for each PE router participating in the routing instance. Although you can use the same route distinguisher on all PE routers for the same VPN routing instance (except for Layer 2 VPNs and VPLS), if you use a unique route distinguisher, you can determine the CE router from which a route originated within the VPN.

To configure a route distinguisher on a PE router, include the `route-distinguisher` statement:

```
route-distinguisher (as-number:number | ip-address:number);
```

For a list of hierarchy levels at which you can include this statement, see the statement summary section for this statement.

The route distinguisher is a 6-byte value that you can specify in one of the following formats:

- *as-number:number*, where *as-number* is an autonomous system (AS) number (a 2-byte value) and *number* is any 4-byte value. The AS number can be in the range 1 through 65,535. We recommend that you use an Internet Assigned Numbers Authority (IANA)-assigned, nonprivate AS number, preferably the Internet service provider's (ISP's) own or the customer's own AS number.

- *ip-address:number*, where *ip-address* is an IP address (a 4-byte value) and *number* is any 2-byte value. The IP address can be any globally unique unicast address. We recommend that you use the address that you configure in the `router-id` statement, which is a nonprivate address in your assigned prefix range.

Configuring Automatic Route Distinguishers

If you configure the `route-distinguisher-id` statement at the `[edit routing-options]` hierarchy level, a route distinguisher is automatically assigned to the routing instance. If you also configure the `route-distinguisher` statement in addition to the `route-distinguisher-id` statement, the value configured for `route-distinguisher` supersedes the value generated from `route-distinguisher-id`.

To assign a route distinguisher automatically, include the `route-distinguisher-id` statement:

```
route-distinguisher-id ip-address;
```

You can include this statement at the following hierarchy levels:

- `[edit routing-options]`
- `[edit logical-systems logical-system-name routing-options]`

A type 1 route distinguisher is automatically assigned to the routing instance using the format *ip-address*:*number*. The IP address is specified by the `route-distinguisher-id` statement and the number is unique for the routing instance.

Configuring Virtual-Router Routing Instances in VPNs

IN THIS SECTION

A virtual-router routing instance, like a VRF routing instance, maintains separate routing and forwarding tables for each instance. However, many of the configuration steps required for VRF routing instances are not required for virtual-router routing instances. Specifically, you do not need to configure a route distinguisher, a routing table policy (the `vrf-export`, `vrf-import`, and `route-distinguisher` statements), or MPLS between the service provider routers.

Configure a virtual-router routing instance by including the following statements:

```
description text;
instance-type virtual-router;
```

```
interface interface-name;
protocols { ... }
```

You can include these statements at the following hierarchy levels:

- `[edit routing-instances routing-instance-name]`
- `[edit logical-systems logical-system-name routing-instances routing-instance-name]`

The following sections explain how to configure a virtual-router routing instance:

Configuring a Routing Protocol Between the Service Provider Routers

The service provider routers need to be able to exchange routing information. You can configure the following protocols for the virtual-router routing instance `protocols` statement configuration at the `[edit routing-instances routing-instance-name]` hierarchy level:

- BGP
- IS-IS
- LDP
- OSPF
- Protocol Independent Multicast (PIM)
- RIP

You can also configure static routes.

IBGP route reflection is not supported for virtual-router routing instances.

If you configure LDP under a virtual-router instance, LDP routes are placed by default in the routing instance’s inet.0 and inet.3 routing tables (for example, sample.inet.0 and sample.inet.3). To restrict LDP routes to only the routing instance’s inet.3 table, include the `no-forwarding` statement:

```
no-forwarding;
```

You can include this statement at the following hierarchy levels:

- `[edit routing-instances routing-instance-name protocols ldp]`
- `[edit logical-systems logical-system-name routing-instances routing-instance-name protocols ldp]`

When you restrict the LDP routes to only the inet.3 routing table, the corresponding IGP route in the inet.0 routing table can be redistributed and advertised into other routing protocols.

For information about routing tables, see Understanding Junos OS Routing Tables.

Configuring Logical Interfaces Between Participating Routers

You must configure an interface to each customer router participating in the routing instance and to each P router participating in the routing instance. Each virtual-router routing instance requires its own separate logical interfaces to all P routers participating in the instance. To configure interfaces for virtual-router instances, include the `interface` statement:

```
interface interface-name;
```

You can include this statement at the following hierarchy levels:

- `[edit routing-instances routing-instance-name]`
- `[edit logical-systems logical-system-name routing-instances routing-instance-name]`

Specify both the physical and logical portions of the interface name, in the following format:

```
physical.logical
```

For example, in `at-1/2/1.2`, `at-1/2/1` is the physical portion of the interface name and `2` is the logical portion. If you do not specify the logical portion of the interface name, `0` is set by default.

You must also configure the interfaces at the `[edit interfaces]` hierarchy level.

One method of providing this logical interface between the provider routers is by configuring tunnels between them. You can configure IP Security (IPsec), generic routing encapsulation (GRE), or IP-IP tunnels between the provider routers, terminating the tunnels at the virtual-router instance.

For information about how to configure tunnels and interfaces, see the Junos OS Services Interfaces Library for Routing Devices.

Configuring Path MTU Checks for VPN Routing Instances

IN THIS SECTION

By default, the maximum transmission unit (MTU) check for VPN routing instances is disabled on M Series routers (except the M320 router) and enabled for the M320 router. On M Series routers, you can configure path MTU checks on the outgoing interfaces for unicast traffic routed on VRF routing instances and on virtual-router routing instances.

When you enable an MTU check, the routing platform sends an Internet Control Message Protocol (ICMP) message when a packet traversing the routing instance exceeds the MTU size and has the `do-not-fragment` bit set. The ICMP message uses the VRF local address as its source address.

For an MTU check to work in a routing instance, you must both include the `vrf-mtu-check` statement at the `[edit chassis]` hierarchy level and assign at least one interface containing an IP address to the routing instance.

For more information about the path MTU check, see the Junos OS Administration Library for Routing Devices.

To configure path MTU checks, do the tasks described in the following sections:

Enabling Path MTU Checks for a VPN Routing Instance

To enable path checks on the outgoing interface for unicast traffic routed on a VRF or virtual-router routing instance, include the `vrf-mtu-check` statement at the `[edit chassis]` hierarchy level:

```
[edit chassis]
vrf-mtu-check;
```

Assigning an IP Address to the VPN Routing Instance

To ensure that the path MTU check functions properly, at least one IP address must be associated with each VRF or virtual-router routing instance. If an IP address is not associated with the routing instance, ICMP reply messages cannot be sent.

Typically, the VRF or virtual-router routing instance IP address is drawn from among the IP addresses associated with interfaces configured for that routing instance. If none of the interfaces associated with a VRF or virtual-router routing instance is configured with an IP address, you need to explicitly configure a logical loopback interface with an IP address. This interface must then be associated with the routing instance. See "Configuring Logical Units on the Loopback Interface for Routing Instances in Layer 3 VPNs" on page 20 for details.

RELATED DOCUMENTATION

Routing Policies, Firewall Filters, and Traffic Policers User Guide

Configuring Policies for the VRF Table on PE Routers in VPNs | 73

Configuring BGP Route Target Filtering for VPNs | 88

Example: Configure a Basic MPLS-Based Layer 3 VPN | 257

Example: Configure MPLS-Based Layer 2 VPNs

Creating Unique VPN Routes Using VRF Tables

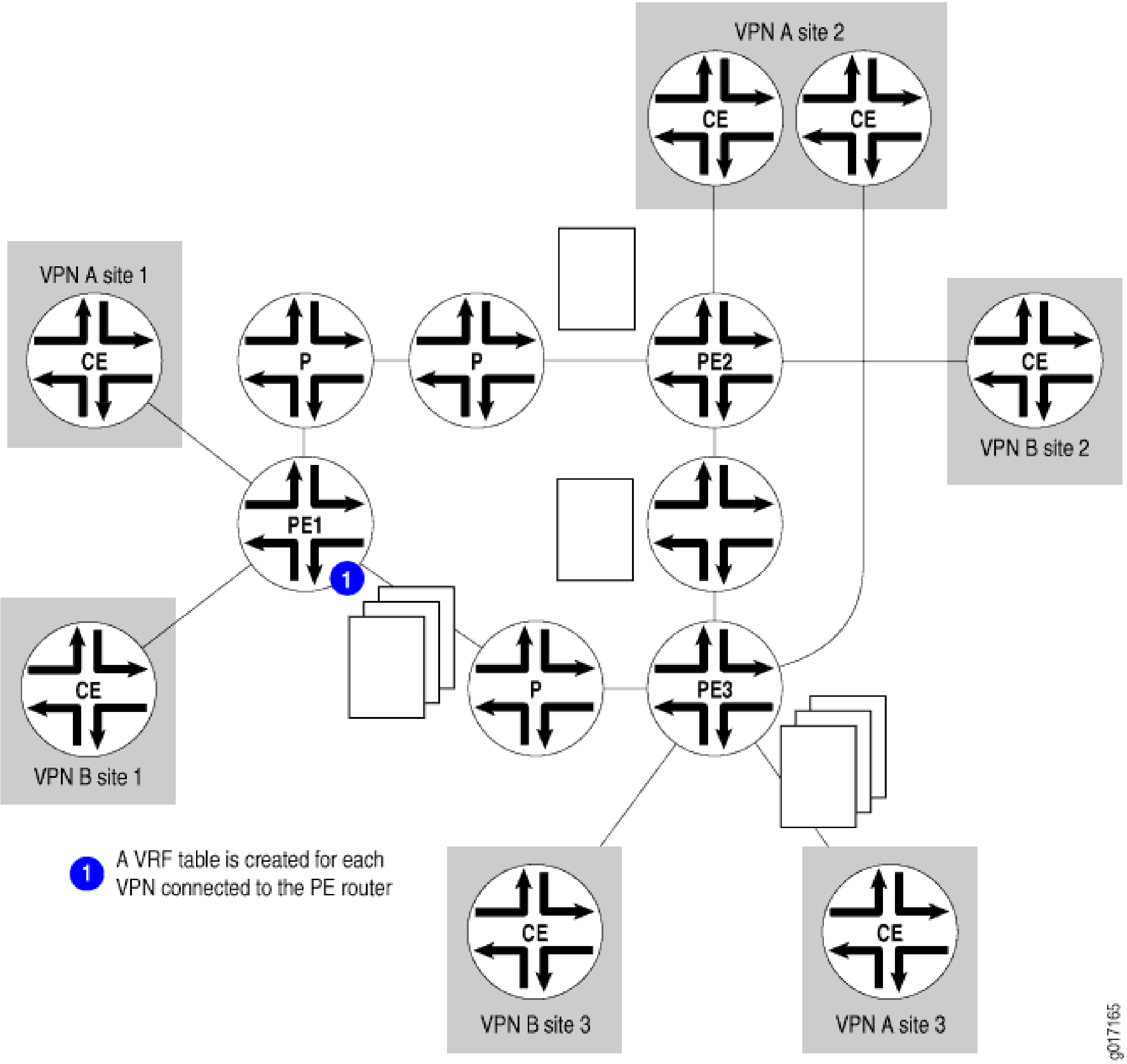
IN THIS SECTION

Understanding Virtual Routing and Forwarding Tables

To separate a VPN's routes from routes in the public Internet or those in other VPNs, the PE router creates a separate routing table for each VPN, called a VPN routing and forwarding (VRF) table. The PE router creates one VRF table for each VPN that has a connection to a CE router. Any customer or site that belongs to the VPN can access only the routes in the VRF tables for that VPN.

Figure 7 on page 33 illustrates the VRF tables that are created on the PE routers. The three PE routers have connections to CE routers that are in two different VPNs, so each PE router creates two VRF tables, one for each VPN.

Figure 7: VRF Tables



Each VRF table is populated from routes received from directly connected CE sites associated with that VRF routing instance and from routes received from other PE routers that passed BGP community filtering and are in the same VPN.

Each PE router also maintains one global routing table (**inet.0**) to reach other routers in and outside the provider's core network.

Each customer connection (that is, each *logical interface*) is associated with one VRF table. Only the VRF table associated with a customer site is consulted for packets from that site.

You can configure the router so that if a next hop to a destination is not found in the VRF table, the router performs a lookup in the global routing table, which is used for Internet access.

The Junos OS uses the following routing tables for VPNs:

- **bgp.l3vpn.0**—Stores routes learned from other PE routers. Routes in the **bgp.l3vpn.0** routing table are copied into a Layer 3 VRF when there is a matching VRF import policy in the PE router. This table is present only on PE routers, and it does not store routes received from directly connected CE routers.

  When a PE router receives a route from another PE router, it places the route into its **bgp.l3vpn.0** routing table. The route is resolved using the information in the **inet.3** routing table. The resultant route is converted into IPv4 format and redistributed to all ***routing-instance-name*.inet.0** routing tables on the PE router if it matches the VRF import policy.

  The **bgp.l3vpn.0** table is also used to resolve routes over the MPLS tunnels that connect the PE routers. These routes are stored in the **inet.3** routing table. PE-to-PE router connectivity must exist in **inet.3** (not just in **inet.0**) for VPN routes to be resolved properly.

  When a router is advertising non-local VPN-IPv4 unicast routes and the router is a route reflector or is performing external peering, the VPN-IPv4 unicast routes are automatically exported into the VPN routing table (**bgp.l3vpn.0**). This enables the router to perform path selection and advertise from the **bgp.l3vpn.0** routing table.

  To determine whether to add a route to the **bgp.l3vpn.0** routing table, the Junos OS checks it against the VRF instance import policies for all the VPNs configured on the PE router. If the VPN-IPv4 route matches one of the policies, it is added to the **bgp.l3vpn.0** routing table. To display the routes in the **bgp.l3vpn.0** routing table, use the **show route table bgp.l3vpn.0** command.

- ***routing-instance-name*.inet.0**—Stores all unicast IPv4 routes received from directly connected CE routers in a routing instance (that is, in a single VPN) and all explicitly configured static routes in the routing instance. This is the VRF table and is present only on PE routers. For example, for a routing instance named **VPN-A**, the routing table for that instance is named **VPN-A.inet.0**.

  When a CE router advertises to a PE router, the PE router places the route into the corresponding ***routing-instance-name*.inet.0** routing table and advertises the route to other PE routers if it passes a VRF export policy. Among other things, this policy tags the route with the route distinguisher (route target) that corresponds to the VPN site to which the CE belongs. A label is also allocated and distributed with the route. The **bgp.l3vpn.0** routing table is not involved in this process.

  The ***routing-instance-name*.inet.0** table also stores routes announced by a remote PE router that match the VRF import policy for that VPN. The PE router redistributed these routes from its **bgp.l3vpn.0** table.

  Routes are not redistributed from the ***routing-instance-name*.inet.0** table to the **bgp.l3vpn.0** table; they are directly advertised to other PE routers.

  For each ***routing-instance-name*.inet.0** routing table, one forwarding table is maintained in the router's Packet Forwarding Engine. This table is maintained in addition to the forwarding tables that correspond to the router's **inet.0** and **mpls.0** routing tables. As with the **inet.0** and **mpls.0** routing tables, the best routes from the ***routing-instance-name*.inet.0** routing table are placed into the forwarding table.

To display the routes in the ***routing-instance-name*.inet.0** table, use the **show route table *routing-instance-name*.inet.0** command.

- **inet.3**—Stores all MPLS routes learned from LDP and RSVP signaling done for VPN traffic. The routing table stores the MPLS routes only if the **traffic-engineering bgp-igp** option is not enabled.

  For VPN routes to be resolved properly, the **inet.3** table must contain routes to all the PE routers in the VPN.

  To display the routes in the **inet.3** table, use the **show route table inet.3** command.

- **inet.0**—Stores routes learned by the IBGP sessions between the PE routers. To provide Internet access to the VPN sites, configure the ***routing-instance-name*.inet.0** routing table to contain a default route to the **inet.0** routing table.

  To display the routes in the **inet.0** table, use the **show route table inet.0** command.

The following routing policies, which are defined in VRF import and export statements, are specific to VRF tables.

- Import policy—Applied to VPN-IPv4 routes learned from another PE router to determine whether the route should be added to the PE router's **bgp.l3vpn.0** routing table. Each routing instance on a PE router has a VRF import policy.

- Export policy—Applied to VPN-IPv4 routes that are announced to other PE routers. The VPN-IPv4 routes are IPv4 routes that have been announced by locally connected CE routers.

VPN route processing differs from normal BGP route processing in one way. In BGP, routes are accepted if they are not explicitly rejected by import policy. However, because many more VPN routes are expected, the Junos OS does not accept (and hence store) VPN routes unless the route matches at least one VRF import policy. If no VRF import policy explicitly accepts the route, it is discarded and not even stored in the **bgp.l3vpn.0** table. As a result, if a VPN change occurs on a PE router—such as adding a new VRF table or changing a VRF import policy—the PE router sends a BGP route refresh message to the other PE routers (or to the route reflector if this is part of the VPN topology) to retrieve all VPN routes so they can be reevaluated to determine whether they should be kept or discarded.

SEE ALSO

> IGP Shortcuts and VPNs

Understanding VRF Localization in Layer 3 VPNs

In a Layer 3 VPN, to separate routes of a VPN from routes in the public Internet or those in other VPNs, the PE router creates a separate routing table for each VPN, called a virtual routing and forwarding (VRF) table. Each VRF uses a route distinguisher and route target to differentiate other VPNs so that each VRF achieves a VPN in a public network. The PE router creates one VRF table for each VPN that has a connection to a CE router. Any customer or site that belongs to the VPN can access only the routes in the VRF tables for that VPN.

The PE routers in a Layer 3 VPN deployment have two types of line cards hosting the following interfaces:

- CE-facing interfaces
- Core-facing interfaces

NOTE: An FPC can be either core-facing or CE-facing.

The VRFs are present on these line cards and currently, in Junos OS, all the routes of all the VRFs are present on all line cards along with chained composite next hops on all the FPCs. This uses up the memory in each line card. Since traffic from CE-facing interfaces comes in only through the corresponding CE-facing FPCs, all the routes and next hops need not be present on all the line cards. VRF localization provides a mechanism for localizing routes of VRF to specific line cards to help maximize the number of routes that a router can handle. CE-facing interfaces localize all the routes of instance type VRF to a specific line card. If CE-facing interfaces are logical interfaces like AE or RLSQ or IRB, then a line card number has to be configured to localize routes. Core-facing line cards store all the VRF routes. These cards have to be configured as VPN core-facing default or VPN core-facing only. Core-facing line cards store routes of all the VRFs, and they are of the following types:

- vpn-core-facing-default — The core-facing FPC installs all the routes and next hops of the VRF routes.
- vpn-core-facing-only — The core-facing FPC installs all the routes and does not store next hops of the VRF routes.

NOTE: Core-facing FPCs can be configured as either core-facing-default or core-facing-only.

Maximizing VPN Routes Using VRF Localization for Layer 3 VPNs

Virtual routing and forwarding (VRF) localization provides a mechanism for localizing routes of VRF to specific line cards to help maximize the number of routes that a router can handle. CE-facing interfaces localize all the routes of instance type VRF to a specific line card. If the CE-facing interfaces are logical interfaces like AE/RLSQ/IRB, then the line card has to be configured to localize routes. Core-facing line cards store all the VRF routes. These cards have to be configured as VPN core-facing only or VPN core-facing default. To configure VRF localization, configure the `localized-fib` statement at the `[edit routing-instances instance-name routing-options]` hierarchy level and configure the `vpn-localization` statement at the `[edit chassis fpc fpc-slot]` hierarchy level. The `show route vpn-localization` command displays the localization information of all the VRFs in the system.

Before you begin to localize the VRF table:

- Configure the interfaces.
- Configure the routing and signaling protocols.

To configure VRF localization:

1. Configure the chassis of the router.
   a. Configure the FPC slot as either VPN core-facing only or VPN core-facing default to store the VRF routes.

   ```
   [edit chassis]
   user@host# set fpc slot-number vpn-localization vpn-core-facing-only
   user@host# set fpc slot-number vpn-localization vpn-core-facing-default
   ```

2. Configure enhanced IP network service on the chassis.

   ```
   [edit chassis]
   user@host# set network-services enhanced-ip
   ```

3. Create an instance type, configure the route distinguisher, and configure the VRF target community and VRF target label.

   ```
   [edit routing-instances routing-instance]
   user@host# set instance-type vrf
   user@host# set interface interface-name
   user@host# set route-distinguisher route-distinguisher-id
   user@host# set provider-tunnel rsvp-te static-lsp vpn1-p2mp
   user@host# set vrf-target vrf-target-community
   user@host# set vrf-table-label
   ```

4. Configure the multipath routing option to balance load independent of the protocol.

```
[edit routing-instances routing-instance routing-options]
user@host# set multipath
```

5. Configure the specific FPC of CE-facing physical interfaces or specify the FPC slot number if the CE-facing interfaces are logical interfaces like AE or RSQL or IRB to localize the VRF routing instance routes.

   - Configure the specific FPC of CE-facing physical interfaces to localize the VRF routing instance routes.

     ```
     [edit routing-instances routing-instance routing-options]
     user@host# set localized-fib
     ```

   - Configure the FPC slot number of the CE-facing logical interfaces like AE or RSQL or IRB to localize the VRF routing instance routes.

     ```
     [edit routing-instances routing-instance routing-options]
     user@host# set localized-fib fpc-slot fpc-slot-number
     ```

6. Configure the peer group of the BGP protocol for the routing instance.

```
[edit routing-instances routing-instance protocols bgp group group-name]
user@host# set type external
user@host# set export direct
user@host# set peer-as 100
user@host# set neighbor IP-address family inet unicast
user@host# set neighbor IP-address family inet6 unicast
```

7. Configure the MVPN protocol for the routing instance.

```
[edit routing-instances routing-instance protocols]
user@host# set mvpn
```

Example: Improving Scalability Using VRF Localization for Layer 3 VPNs

IN THIS SECTION

This example shows how to configure VRF localization on MX Series routers, which enables you to improve the VPN scalability on MX Series routers.

Requirements

This example uses the following hardware and software components:

- Five MX Series 5G Universal Routing Platforms
- Junos OS Release 14.2 or later running on all devices

Before you begin:

1. Configure the device interfaces.
2. Configure the BGP protocol.

Overview

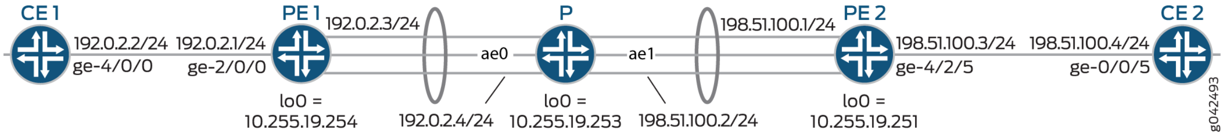
IN THIS SECTION

Starting with Junos OS Release 14.2, the VRF localization provides a mechanism for localizing routes of VRF to specific line cards which helps maximize the number of routes that a router can handle. CE-facing interfaces localize all the routes of instance type VRF to a specific line card. If the CE-facing interfaces are logical interfaces like AE or RLSQ or IRB, then the line card has to be configured to localize routes. Core-facing line cards store all the VRF routes. These cards have to be configured as VPN core-facing only or VPN core-facing default. To configure VRF localization, configure the `localized-fib` configuration statement at the `[edit routing-instances` *`instance-name`* `routing-options]` hierarchy level and configure `vpn-localization` at the `[edit chassis fpc` *`fpc-slot`*`]` hierarchy level. The `show route vpn-localization` command displays the localization information of all the VRFs in the system.

Topology

In the topology shown in Figure 8 on page 40, VRF localization is configured on Device PE1.

Figure 8: Example VRF Localization



Configuration

IN THIS SECTION

CLI Quick Configuration

To quickly configure this example, copy the following commands, paste them into a text file, remove any line breaks, change any details necessary to match your network configuration, copy and paste the commands into the CLI at the `[edit]` hierarchy level, and then enter `commit` from the configuration mode.

CE1

```
set interfaces ge-4/0/0 unit 0 family inet address 192.0.2.2/24
set interfaces ge-4/0/0 unit 0 family inet6 address abcd:a:a:a:1::2/126
set protocols bgp group vpn1 type external
set protocols bgp group vpn1 export direct
```

```
set protocols bgp group vpn1 peer-as 10
set protocols bgp group vpn1 neighbor 192.0.2.1 family inet unicast
set protocols bgp group vpn1 neighbor abcd:a:a:a:1::1 family inet6 unicast
set policy-options policy-statement direct from protocol direct
set policy-options policy-statement direct then accept
set policy-options policy-statement load-balancing-policy then load-balance per-packet
set routing-options autonomous-system 100
set routing-options forwarding-table export load-balancing-policy
```

PE1

```
set chassis redundancy graceful-switchover
set chassis aggregated-devices ethernet device-count 16
set chassis fpc 8 vpn-localization vpn-core-facing-only
set chassis network-services enhanced-ip
set interfaces ge-2/0/0 unit 0 family inet address 192.0.2.1/24
set interfaces ge-2/0/0 unit 0 family inet6 address abcd:a:a:a:1::1/126
set interfaces ge-8/1/0 gigether-options 802.3ad ae0
set interfaces ge-8/1/9 gigether-options 802.3ad ae0
set interfaces ae0 unit 0 family inet address 192.0.2.3/24
set interfaces ae0 unit 0 family iso
set interfaces ae0 unit 0 family mpls
set interfaces lo0 unit 1 family inet address 10.255.19.254/24
set interfaces lo0 unit 1 family inet6 address abcd::10:0:1:1/128
set policy-options policy-statement direct from protocol direct
set policy-options policy-statement direct then accept
set policy-options policy-statement load-balancing-policy then load-balance per-packet
set protocols rsvp interface ae0.0
set protocols mpls ipv6-tunneling
set protocols mpls icmp-tunneling
set protocols mpls label-switched-path pe1-pe2-p2mp-1 from 10.255.19.254
set protocols mpls label-switched-path pe1-pe2-p2mp-1 to 10.255.19.251
set protocols mpls label-switched-path pe1-pe2-p2mp-1 link-protection
set protocols mpls label-switched-path pe1-pe2-p2mp-1 p2mp vpn1-p2mp
set protocols mpls label-switched-path pe1-pe3-p2mp-1 from 10.255.19.254
set protocols mpls label-switched-path pe1-pe3-p2mp-1 to 10.255.19.203
set protocols mpls label-switched-path pe1-pe3-p2mp-1 link-protection
set protocols mpls label-switched-path pe1-pe3-p2mp-1 p2mp vpn1-p2mp
set protocols mpls interface ae0.0
set protocols bgp group mpbg type internal
set protocols bgp group mpbg local-address 10.255.19.254
set protocols bgp group mpbg family inet unicast
```

```
set protocols bgp group mpbg family inet-vpn unicast
set protocols bgp group mpbg family inet6 unicast
set protocols bgp group mpbg family inet6-vpn unicast
set protocols bgp group mpbg family inet-mvpn signaling
set protocols bgp group mpbg family inet6-mvpn signaling
set protocols bgp group mpbg neighbor 10.255.19.253
set protocols ospf traffic-engineering
set protocols ospf area 0.0.0.0 interface ae0.0
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set protocols ldp interface ae0.0
set routing-instances vpn1 instance-type vrf
set routing-instances vpn1 interface ge-2/0/0.0
set routing-instances vpn1 interface lo0.1
set routing-instances vpn1 route-distinguisher 1:1
set routing-instances vpn1 provider-tunnel rsvp-te static-lsp vpn1-p2mp
set routing-instances vpn1 vrf-target target:1:1
set routing-instances vpn1 vrf-table-label
set routing-instances vpn1 routing-options multipath
set routing-instances vpn1 routing-options localized-fib
set routing-instances vpn1 protocols bgp group grp1 type external
set routing-instances vpn1 protocols bgp group grp1 export direct
set routing-instances vpn1 protocols bgp group grp1 peer-as 100
set routing-instances vpn1 protocols bgp group grp1 neighbor 192.0.2.2 family inet unicast
set routing-instances vpn1 protocols bgp group grp1 neighbor abcd:a:a:a:1::2 family inet6 unicast
set routing-instances vpn1 protocols mvpn
set routing-options nonstop-routing
set routing-options autonomous-system 10
set routing-options forwarding-table export load-balancing-policy
set routing-options forwarding-table chained-composite-next-hop ingress l3vpn extended-space
```

P

```
set chassis aggregated-devices ethernet device-count 16
set interfaces ge-1/0/1 gigether-options 802.3ad ae0
set interfaces ge-1/0/3 gigether-options 802.3ad ae0
set interfaces ge-1/1/1 gigether-options 802.3ad ae1
set interfaces ae0 unit 0 family inet address 192.0.2.4/24
set interfaces ae0 unit 0 family iso
set interfaces ae0 unit 0 family mpls
set interfaces ae1 unit 0 family inet address 198.51.100.2/24
set interfaces ae1 unit 0 family iso
set interfaces ae1 unit 0 family mpls
```

```
set routing-options autonomous-system 10
set routing-options forwarding-table export load-balancing-policy
set protocols rsvp interface ae0.0
set protocols rsvp interface ae1.0
set protocols mpls ipv6-tunneling
set protocols mpls icmp-tunneling
set protocols mpls interface ae0.0
set protocols mpls interface ae1.0
set protocols bgp group mpbg type internal
set protocols bgp group mpbg local-address 10.255.19.253
set protocols bgp group mpbg family inet unicast
set protocols bgp group mpbg family inet-vpn unicast
set protocols bgp group mpbg family inet6 unicast
set protocols bgp group mpbg family inet6-vpn unicast
set protocols bgp group mpbg family inet-mvpn signaling
set protocols bgp group mpbg family inet6-mvpn signaling
set protocols bgp group mpbg cluster 10.255.19.253
set protocols bgp group mpbg neighbor 10.255.19.254
set protocols bgp group mpbg neighbor 10.255.19.251
set protocols bgp group mpbg neighbor 10.255.19.203
set protocols ospf traffic-engineering
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set protocols ospf area 0.0.0.0 interface ae0.0
set protocols ospf area 0.0.0.0 interface ae1.0
set protocols ldp interface ae0.0
set protocols ldp interface ae1.0
set policy-options policy-statement load-balancing-policy then load-balance per-packet
```

PE2

```
set chassis redundancy graceful-switchover
set chassis aggregated-devices ethernet device-count 16
set interfaces ge-4/2/1 gigether-options 802.3ad ae1
set interfaces ge-4/2/5 unit 0 family inet address 198.51.100.3/24
set interfaces ge-4/2/5 unit 0 family inet6 address abcd:a:a:a:2::1/126
set interfaces ae1 unit 0 family inet address 198.51.100.1/24
set interfaces ae1 unit 0 family iso
set interfaces ae1 unit 0 family mpls
set interfaces lo0 unit 2 family inet address 10.255.19.251/24
set interfaces lo0 unit 2 family inet6 address abcd::203:0:113:2/128
set policy-options policy-statement direct from protocol direct
set policy-options policy-statement direct then accept
```

```
set policy-options policy-statement load-balancing-policy then load-balance per-packet
set protocols rsvp interface ae1.0
set protocols mpls ipv6-tunneling
set protocols mpls icmp-tunneling
set protocols mpls label-switched-path pe2-pe1-p2mp-1 from 10.255.19.251
set protocols mpls label-switched-path pe2-pe1-p2mp-1 to 10.255.19.254
set protocols mpls label-switched-path pe2-pe1-p2mp-1 link-protection
set protocols mpls label-switched-path pe2-pe1-p2mp-1 p2mp vpn1-p2mp
set protocols mpls label-switched-path pe2-pe3-p2mp-1 from 10.255.19.251
set protocols mpls label-switched-path pe2-pe3-p2mp-1 to 10.255.19.203
set protocols mpls label-switched-path pe2-pe3-p2mp-1 link-protection
set protocols mpls label-switched-path pe2-pe3-p2mp-1 p2mp vpn1-p2mp
set protocols mpls interface ae1.0
set protocols bgp group mpbg type internal
set protocols bgp group mpbg local-address 10.255.19.251
set protocols bgp group mpbg family inet unicast
set protocols bgp group mpbg family inet-vpn unicast per-prefix-label
set protocols bgp group mpbg family inet6 unicast
set protocols bgp group mpbg family inet6-vpn unicast per-prefix-label
set protocols bgp group mpbg family inet-mvpn signaling
set protocols bgp group mpbg family inet6-mvpn signaling
set protocols bgp group mpbg neighbor 10.255.19.253
set protocols ospf traffic-engineering
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set protocols ospf area 0.0.0.0 interface ae1.0
set protocols ldp interface ae1.0
set routing-instances vpn1 instance-type vrf
set routing-instances vpn1 interface ge-4/2/5.0
set routing-instances vpn1 route-distinguisher 1:1
set routing-instances vpn1 provider-tunnel rsvp-te static-lsp vpn1-p2mp
set routing-instances vpn1 vrf-target target:1:1
set routing-instances vpn1 vrf-table-label
set routing-instances vpn1 routing-options multipath
set routing-instances vpn1 protocols bgp group grp1 type external
set routing-instances vpn1 protocols bgp group grp1 export direct
set routing-instances vpn1 protocols bgp group grp1 peer-as 200
set routing-instances vpn1 protocols bgp group grp1 neighbor 198.51.100.4 family inet unicast
set routing-instances vpn1 protocols bgp group grp1 neighbor abcd:a:a:a:2::2 family inet6 unicast
set routing-instances vpn1 protocols mvpn
set routing-options nonstop-routing
set routing-options autonomous-system 10
set routing-options forwarding-table export load-balancing-policy
```

CE2

```
set interfaces ge-0/0/5 unit 0 family inet address 198.51.100.4/24
set interfaces ge-0/0/5 unit 0 family inet6 address abcd:a:a:a:2::2/126
set protocols bgp group vpn1 type external
set protocols bgp group vpn1 export direct
set protocols bgp group vpn1 export vpn1
set protocols bgp group vpn1 peer-as 10
set protocols bgp group vpn1 neighbor 198.51.100.3 family inet unicast
set protocols bgp group vpn1 neighbor abcd:a:a:a:2::1 family inet6 unicast
set policy-options policy-statement direct from protocol direct
set policy-options policy-statement direct then accept
set policy-options policy-statement load-balancing-policy then load-balance per-packet
set routing-options autonomous-system 200
set routing-options forwarding-table export load-balancing-policy
```

Configuring Device PE1

Step-by-Step Procedure

The following example requires you to navigate various levels in the configuration hierarchy. For information about navigating the CLI, see *Using the CLI Editor in Configuration Mode*.

To configure Device PE1:

1. Specify the number of aggregated Ethernet interfaces to be created, configure the FPCs as vpn-core-facing-only, and enable enhanced IP network services.

```
[edit chassis]
user@PE1# set redundancy graceful-switchover
user@PE1# set aggregated-devices ethernet device-count 16
user@PE1# set fpc 8 vpn-localization vpn-core-facing-only
user@PE1# set network-services enhanced-ip
```

2. Configure the interfaces.

```
[edit interfaces]
user@PE1# set ge-2/0/0 unit 0 family inet address 192.0.2.1/24
user@PE1# set ge-2/0/0 unit 0 family inet6 address abcd:a:a:a:1::1/126
user@PE1# set ge-8/1/0 gigether-options 802.3ad ae0
```

```
user@PE1# set ge-8/1/9 gigether-options 802.3ad ae0
user@PE1# set ae0 unit 0 family inet address 192.0.2.3/24
user@PE1# set ae0 unit 0 family iso
user@PE1# set ae0 unit 0 family mpls
user@PE1# set lo0 unit 1 family inet address 10.255.19.254/24
user@PE1# set lo0 unit 1 family inet6 address abcd::10:0:1:1/128
```

3. Configure policy options to load balance the packets.

```
[edit policy-options policy-statement]
user@PE1# set direct from protocol direct
user@PE1# set direct then accept
user@PE1# set load-balancing-policy then load-balance per-packet
```

4. Configure the RSVP protocol on the interface.

```
[edit protocols rsvp]
user@PE1# set interface ae0.0
```

5. Configure the MPLS protocol.

```
[edit protocols mpls]
user@PE1# set ipv6-tunneling
user@PE1# set icmp-tunneling
user@PE1# set label-switched-path pe1-pe2-p2mp-1 from 10.255.19.254
user@PE1# set label-switched-path pe1-pe2-p2mp-1 to 10.255.19.251
user@PE1# set label-switched-path pe1-pe2-p2mp-1 link-protection
user@PE1# set label-switched-path pe1-pe2-p2mp-1 p2mp vpn1-p2mp
user@PE1# set label-switched-path pe1-pe3-p2mp-1 from 10.255.19.254
user@PE1# set label-switched-path pe1-pe3-p2mp-1 to 10.255.19.203
user@PE1# set label-switched-path pe1-pe3-p2mp-1 link-protection
user@PE1# set label-switched-path pe1-pe3-p2mp-1 p2mp vpn1-p2mp
user@PE1# set interface ae0.0
```

6. Configure the BGP protocol for the mpbg group.

```
[edit protocols bgp group mpbg]
user@PE1# set type internal
user@PE1# set local-address 10.255.19.254
```

```
user@PE1# set family inet unicast
user@PE1# set family inet-vpn unicast
user@PE1# set family inet6 unicast
user@PE1# set family inet6-vpn unicast
user@PE1# set family inet-mvpn signaling
user@PE1# set family inet6-mvpn signaling
user@PE1# set neighbor 10.255.19.253
```

7. Configure the OSPF protocol.

```
[edit protocols ospf]
user@PE1# set traffic-engineering
user@PE1# set area 0.0.0.0 interface ae0.0
user@PE1# set area 0.0.0.0 interface lo0.0 passive
```

8. Configure the LDP protocol on the interface.

```
[edit protocols]
user@PE1# set ldp interface ae0.0
```

9. Create an instance type and configure the routing instances on the interface.

```
[edit routing-instances vpn1]
user@PE1# set instance-type vrf
user@PE1# set interface ge-2/0/0.0
user@PE1# set interface lo0.1
```

10. Configure the route distinguisher, and configure the static LSP for the provider tunnel RSVP-TE.

```
[edit routing-instances vpn1]
user@PE1# set route-distinguisher 1:1
user@PE1# set provider-tunnel rsvp-te static-lsp vpn1-p2mp
```

11. Configure the VRF target and the VRF target label for the routing instance.

```
[edit routing-instances vpn1]
user@PE1# set vrf-target target:1:1
user@PE1# set vrf-table-label
```

12. Configure the multipath routing option for a routing instance, and configure the localized fib routing option for the routing instance.

```
[edit routing-instances vpn1 routing-options]
user@PE1# set multipath
user@PE1# set localized-fib
```

13. Configure the group of BGP protocols for a routing instance.

```
[edit routing-instances vpn1 protocols bgp group grp1]
user@PE1# set type external
user@PE1# set export direct
user@PE1# set peer-as 100
user@PE1# set neighbor 192.0.2.2 family inet unicast
user@PE1# set neighbor abcd:a:a:a:1::2 family inet6 unicast
```

14. Configure the MVPN protocols.

```
[edit routing-instances vpn1]
user@PE1# set protocols mvpn
```

15. Configure the nonstop active routing and the autonomous system number for a routing option.

```
[edit routing-options]
user@PE1# set nonstop-routing
user@PE1# set autonomous-system 10
```

16. Configure the load-balancing policy for the forwarding table and extended space for the chained composite next hop for the L3VPN of the forwarding table.

    ```
    [edit routing-options]
    user@PE1# set forwarding-table export load-balancing-policy
    user@PE1# set forwarding-table chained-composite-next-hop ingress l3vpn extended-space
    ```

Results

From configuration mode, confirm your configuration by entering the `show chassis`, `show interfaces`, `show policy-options`, `show protocols`, `show routing-instances`, and `show routing-options` commands. If the output does not display the intended configuration, repeat the instructions in this example to correct the configuration.

```
user@PE1# show chassis
redundancy {
    graceful-switchover;
}
aggregated-devices {
    ethernet {
        device-count 16;
    }
}
fpc 8 {
    vpn-localization vpn-core-facing-only;
}
network-services enhanced-ip;
```

```
user@PE1# show interfaces
ge-2/0/0 {
    unit 0 {
        family inet {
            address 192.0.2.1/24;
        }
        family inet6 {
            address abcd:a:a:a:1::1/126;
        }
    }
}
```

```
ge-8/1/0 {
    gigether-options {
        802.3ad ae0;
    }
}
ge-8/1/9 {
    gigether-options {
        802.3ad ae0;
    }
}
ae0 {
    unit 0 {
        family inet {
            address 192.0.2.3/24;
        }
        family iso;
        family mpls;
    }
}
lo0 {
    unit 1 {
        family inet {
            address 10.255.19.254/24;
        }
        family inet6 {
            address abcd::10:0:1:1/128;
        }
    }
}
```

```
user@PE1# show policy-options
policy-statement direct {
    from protocol direct;
    then accept;
}
policy-statement load-balancing-policy {
    then {
        load-balance per-packet;
```

```
    }
}
```

```
user@PE1# show routing-options
nonstop-routing;
autonomous-system 10;
forwarding-table {
    export load-balancing-policy;
    chained-composite-next-hop {
        ingress {
            l3vpn extended-space;
        }
    }
}
```

```
user@PE1# show routing-instances
vpn1 {
    instance-type vrf;
    interface ge-2/0/0.0;
    interface lo0.1;
    route-distinguisher 1:1;
    provider-tunnel {
        rsvp-te {
            static-lsp vpn1-p2mp;
        }
    }
    vrf-target target:1:1;
    vrf-table-label;
    routing-options {
        multipath;
        localized-fib;
    }
    protocols {
        bgp {
            group grp1 {
                type external;
                export direct;
                peer-as 100;
                neighbor 192.0.2.2 {
                    family inet {
```

```
                    unicast;
                }
            }
            neighbor abcd:a:a:a:1::2 {
                family inet6 {
                    unicast;
                }
            }
        }
    }
    mvpn;
  }
}
```

```
user@PE1# show protocols
rsvp {
    interface ae0.0;
}
mpls {
    ipv6-tunneling;
    icmp-tunneling;
    label-switched-path pe1-pe2-p2mp-1 {
        from 10.255.19.254;
        to 10.255.19.251;
        link-protection;
        p2mp vpn1-p2mp;
    }
    label-switched-path pe1-pe3-p2mp-1 {
        from 10.255.19.254;
        to 10.255.19.203;
        link-protection;
        p2mp vpn1-p2mp;
    }
    interface ae0.0;
}
bgp {
    group mpbg {
        type internal;
        local-address 10.255.19.254;
        family inet {
            unicast;
```

```
        }
        family inet-vpn {
            unicast;
        }
        family inet6 {
            unicast;
        }
        family inet6-vpn {
            unicast;
        }
        family inet-mvpn {
            signaling;
        }
        family inet6-mvpn {
            signaling;
        }
        neighbor 10.255.19.253;
    }
}
ospf {
    traffic-engineering;
    area 0.0.0.0 {
        interface ae0.0;
        interface lo0.0 {
            passive;
        }
    }
}
ldp {
    interface ae0.0;
}
```

If you are done configuring the device, enter `commit` from configuration mode.

Verification

IN THIS SECTION

Confirm that the configuration is working properly.

Verifying VRF Localization

Purpose

Verify the localization of VRF in a Layer 3 VPN.

Action

From operational mode, run the `show route vpn-localization` command for Device PE1.

```
user@PE1> show route vpn-localization

Routing table: vpn1.inet, Localized
  Index: 7, Address Family: inet, Localization status: Complete
  Local FPC's: 2 8

Routing table: vpn1.inet6, Localized
  Index: 7, Address Family: inet6, Localization status: Complete
  Local FPC's: 2 8

Routing table: vpn2.inet, Non-localized
  Index: 8, Address Family: inet, Localization status: Complete
  Local FPC's: All

Routing table: vpn2.inet6, Non-localized
  Index: 8, Address Family: inet6, Localization status: Complete
  Local FPC's: All
```

Meaning

The output shows the localization information of all the VRFs.

Verifying VRF Localization for a VPN

Purpose

Verify VRF localization for a VPN.

Action

From operational mode, run the `show route vpn-localization vpn-name` *`vpn-name`* command.

```
user@PE1> show route vpn-localization vpn-name vpn1

Routing table: vpn1.inet, Localized
  Index: 7, Address Family: inet, Localization status: Complete
  Local FPC's: 2 8

Routing table: vpn1.inet6, Localized
  Index: 7, Address Family: inet6, Localization status: Complete
  Local FPC's: 2 8
```

Meaning

The output shows the VPN localization of a VPN.

Filtering Packets in Layer 3 VPNs Based on IP Headers

IN THIS SECTION

Including the `vrf-table-label` statement in the configuration for a routing instance makes it possible to map the inner label to a specific VRF routing table; such mapping allows the examination of the encapsulated IP header at an egress VPN router. You might want to enable this functionality so that you can do either of the following:

- Forward traffic on a PE-router-to-CE-device interface, in a shared medium, where the CE device is a Layer 2 switch without IP capabilities (for example, a metro Ethernet switch).

  The first lookup is done on the VPN label to determine which VRF table to refer to, and the second lookup is done on the IP header to determine how to forward packets to the correct end hosts on the shared medium.

- Perform egress filtering at the egress PE router.

  The first lookup on the VPN label is done to determine which VRF routing table to refer to, and the second lookup is done on the IP header to determine how to filter and forward packets. You can enable this functionality by configuring output filters on the VRF interfaces.

  When you include the `vrf-table-label` statement in the configuration of a VRF routing table, a label-switched interface (LSI) logical interface label is created and mapped to the VRF routing table. Any routes in such a VRF routing table are advertised with the LSI logical interface label allocated for the VRF routing table. When packets for this VPN arrive on a core-facing interface, they are treated as if the enclosed IP packet arrived on the LSI interface and are then forwarded and filtered based on the correct table.

To filter traffic based on the IP header, include the `vrf-table-label` statement:

```
vrf-table-label {
    source-class-usage;
}
```

You can include the statement at the following hierarchy levels:

- `[edit routing-instances` *`routing-instance-name`*`]`
- `[edit logical-systems` *`logical-system-name`* `routing-instances` *`routing-instance-name`*`]`

You can include the `vrf-table-label` statement for both IPv4 and IPv6 Layer 3 VPNs. If you include the statement for a dual-stack VRF routing table (where both IPv4 and IPv6 routes are supported), the statement applies to both the IPv4 and IPv6 routes and the same label is advertised for both sets of routes.

You can also configure SCU accounting for Layer 3 VPNs configured with the `vrf-table-label` statement by also including the `source-class-usage` option. Include the `source-class-usage` statement at the `[edit routing-instances` *`routing-instance-name`* `vrf-table-label]` hierarchy level. The `source-class-usage` statement at this hierarchy level is supported only for the `vrf` instance type (Layer 3 VPNs). DCU is not supported for the `vrf-table-label` statement. For more information, see Enabling Source Class and Destination Class Usage.

The following sections provide more information about traffic filtering based on the IP header:

Egress Filtering Options

You can enable egress filtering (which allows egress Layer 3 VPN PE routers to perform lookups on the VPN label and IP header at the same time) by including the `vrf-table-label` statement at the `[edit routing-instances` *`instance-name`*`]` hierarchy level. There is no restriction on including this statement for CE-router-to-PE-router interfaces, but there are several limitations on other interface types, as described in subsequent sections in this topic.

You can also enable egress filtering by configuring a VPN tunnel (VT) interface on routing platforms equipped with a Tunnel Services Physical Interface Card (PIC). When you enable egress filtering this way, there is no restriction on the type of core-facing interface used. There is also no restriction on the type of CE-router-to-PE-router interface used.

Support on Aggregated and VLAN Interfaces for IP-Based Filtering

Support for the `vrf-table-label` statement over aggregated and VLAN interfaces is available on the routers summarized in Table 1 on page 57.

Table 1: Support for Aggregated and VLAN Interfaces

Interfaces	M Series Router Without an Enhanced FPC	M Series Router with an Enhanced FPC	M320 Router	T Series Router
Aggregated	No	Yes	Yes	Yes
VLAN	No	Yes	Yes	Yes

NOTE: The `vrf-table-label` statement is not supported for Aggregated Gigabit Ethernet, 10-Gigabit Ethernet, and VLAN physical interfaces on M120 routers.

Support on ATM and Frame Relay Interfaces for IP-Based Filtering

Support for the `vrf-table-label` statement over Asynchronous Transfer Mode (ATM) and Frame Relay interfaces is available on the routers summarized in Table 2 on page 58.

Table 2: Support for ATM and Frame Relay Interfaces

Interfaces	M Series Router Without an Enhanced FPC	M Series Router with an Enhanced FPC	M320 Router	T Series Router
ATM1	No	No	No	No
ATM2 intelligent queuing (IQ)	No	Yes	Yes	Yes
Frame Relay	No	Yes	Yes	Yes
Channelized	No	No	No	No

When you include the `vrf-table-label` statement, be aware of the following limitations with ATM or Frame Relay interfaces:

- The `vrf-table-label` statement is supported on ATM interfaces, but with the following limitations:
  - ATM interfaces can be configured on the M320 router and the T Series routers, and on M Series routers with an enhanced FPC.
  - The interface can only be a PE router interface receiving traffic from a P router.
  - The router must have an ATM2 IQ PIC.
- The `vrf-table-label` statement is also supported on Frame Relay encapsulated interfaces, but with the following limitations:
  - Frame Relay interfaces can be configured on the M320 router and the T Series routers, and on M Series routers with an enhanced FPC.
  - The interface can only be a PE router interface receiving traffic from a P router.

Support on Ethernet, SONET/SDH, and T1/T3/E3 Interfaces for IP-Based Filtering

Support for the `vrf-table-label` statement over Ethernet, SONET/SDH, and T1/T3/E3 interfaces is available on the routers summarized in Table 3 on page 59.

Table 3: Support for Ethernet, SONET/SDH, and T1/T3/E3 Interfaces

Interfaces	M Series Router Without an Enhanced FPC	M Series Router with an Enhanced FPC	M320 Router	T Series Router
Ethernet	Yes	Yes	Yes	Yes
SONET/SDH	Yes	Yes	Yes	Yes
T1/T3/E3	Yes	Yes	Yes	Yes

Only the following Ethernet PICs support the `vrf-table-label` statement on M Series routers without an Enhanced FPC:

- 1-port Gigabit Ethernet
- 2-port Gigabit Ethernet
- 4-port Fast Ethernet

Support on SONET/SDH and DS3/E3 Channelized Enhanced Intelligent Queuing Interfaces for IP-Based Filtering

Support for the `vrf-table-label` statement for the specified channelized IQE interfaces is only available on M120 and M320 routers with Enhanced III FPCs as summarized in Table 4 on page 59.

Table 4: Support for Channelized IQE Interfaces on M320 Routers with Enhanced III FPCs

Interfaces	M120 Routers with Enhanced III FPCs	M320 Routers with Enhanced III FPCs
OC12	Yes	Yes
STM4	Yes	Yes
OC3	Yes	Yes
STM1	Yes	Yes

Table 4: Support for Channelized IQE Interfaces on M320 Routers with Enhanced III FPCs *(Continued)*

Interfaces	M120 Routers with Enhanced III FPCs	M320 Routers with Enhanced III FPCs
DS3	Yes	Yes
E3	Yes	Yes

The following IQE Type-1 PICs are supported:

- 1-port OC12/STM4 IQE with SFP
- 4-port OC3/STM1 IQE with SFP
- 4-port DS3/E3 IQE with BNC
- 2-port Channelized OC3/STM1 IQE with SFP, with no SONET partitions
- 1-port Channelized OC12/STM4 IQE with SFP, with no SONET partitions

The following constraints are applicable with respect to a router configuration utilizing logical systems:

- Multiport IQE PIC interfaces constraints—On multiport IQE PICs, such as the 2-port Channelized OC3/STM1 IQE with SFP, if the port 1 interface is configured as one logical system with its own routing-instance and the port 2 interface is configured as a different logical system with its own routing instances such that there are core-facing logical interfaces on both port 1 and port 2, then you cannot configure the `vrf-table-label` statement on routing-instance in both logical systems. Only one set of LSI labels are supported; the last routing instance with the `vrf-table-label` statement configured is committed.

- Frame Relay encapsulation and logical interfaces across logical systems constraints—Similar to the multiport PIC with logical systems, if you try to configure one logical interface of an IQE PIC with Frame Relay encapsulation in one logical system and configure another logical interface on the same IQE PIC in the second logical system, the configuration will not work for all the `vrf-table-label` statement configured instances. It will only work for the instances configured in one of the logical systems.

Both the above constraints occur because the router configuration maintains one LSI tree in the Packet Forwarding Engine per logical system, which is common across all streams. The stream channel table lookup is then adjusted to point to the LSI tree. In the case of multiport type-1 IQE PICs, all physical interfaces share the same stream. Therefore, the logical interfaces (multiport or not) obviously share the same stream. Consequently, the LSI binding is at the stream level. Hence, provisioning logical interfaces under the same stream provisioned to be core-facing and supporting a different set of routing instances with the `vrf-table-label` statement is not supported.

Support on Multilink PPP and Multilink Frame Relay Interfaces for IP-Based Filtering

Support for the `vrf-table-label` statement over Multilink Point-to-Point Protocol (MLPPP) and Multilink Frame Relay (MLFR) interfaces is available on the routers summarized in Table 5 on page 61.

Table 5: Support for Multilink PPP and Multilink Frame Relay Interfaces

Interfaces	M Series Router Without an Enhanced FPC	M Series Router with an Enhanced FPC	M320	T Series Router	MX Series Router
MLPPP	No	Yes	No	No	No
End-to-End MLFR (FRF.15)	No	Yes	No	No	No
UNI/NNI MLFR (FRF.16)	No	No	No	No	No

M Series routers must have an AS PIC to support the `vrf-table-label` statement over MLPPP and MLFR interfaces. The `vrf-table-label` statement over MLPPP interfaces is not supported on M120 routers.

Support for IP-Based Filtering of Packets with Null Top Labels

You can include the `vrf-table-label` statement in the configuration for core-facing interfaces receiving MPLS packets with a null top label, which might be transmitted by some vendors' equipment. These packets can be received only on the M320 router, the M10i router, and T Series Core routers using one of the following PICs:

- 1-port Gigabit Ethernet with SFP
- 2-port Gigabit Ethernet with SFP
- 4-port Gigabit Ethernet with SFP
- 10-port Gigabit Ethernet with SFP
- 1-port SONET STM4
- 4-port SONET STM4
- 1-port SONET STM16
- 1-port SONET STM16 (non-SFP)

- 4-port SONET STM16
- 1-port SONET STM64

The following PICs can receive packets with null top labels, but only when installed in an M120 router or an M320 router with an Enhanced III FPC:

- 1-port 10-Gigabit Ethernet
- 1-port 10-Gigabit Ethernet IQ2

General Limitations on IP-Based Filtering

The following limitations apply when you include the `vrf-table-label` statement:

- Firewall filters cannot be applied to interfaces included in a routing instance on which you have configured the `vrf-table-label` statement.
- The time-to-live (TTL) value in the MPLS header is not copied back to the IP header of packets sent from the PE router to the CE router.
- You cannot include the `vrf-table-label` statement in a routing instance configuration that also includes a virtual loopback tunnel interface; the commit operation fails in this case.
- When you include the statement, MPLS packets with label-switched interface (LSI) labels that arrive on core-facing interfaces are not counted at the logical interface level if the core-facing interface is any of the following:
  - ATM
  - Frame Relay
  - Ethernet configured with VLANs
  - Aggregated Ethernet configured with VLANs
- For LMNR, Stoli, and I-Chip-based Packet Forwarding Engines, you cannot include the statement in the configuration of a VRF routing instance if the PE-router-to-P-router interface is any of the following interfaces:

  > **NOTE**: The `vrf-table-label` statement is supported when the PE-router-to-P-router interface is a tunnel interface on a Junos Trio-based Packet Forwarding Engine, so no limitation applies.

  - Aggregated SONET/SDH interface

  - Channelized interface
  - Tunnel interface (for example, generic routing encapsulation [GRE] or IP Security [IPsec])
  - Circuit cross-connect (CCC) or translational cross-connect (TCC) encapsulated interface
  - Logical tunnel interface
  - Virtual private LAN service (VPLS) encapsulated interface

  > **NOTE**: All CE-router-to-PE-router and PE-router-to-CE-router interfaces are supported.

- You cannot include the `vrf-table-label` statement in the configuration of a VRF routing instance if the PE-router-to-P-router PIC is one of the following PICs:
  - 10-port E1
  - 8-port Fast Ethernet
  - 12-port Fast Ethernet
  - 48-port Fast Ethernet
  - ATM PIC other than the ATM2 IQ
- Label-switched interface (LSI) traffic statistics are not supported for Intelligent Queuing 2 (IQ2), Enhanced IQ (IQE), and Enhanced IQ2 (IQ2E) PICs on M Series routers.

SEE ALSO

Enabling Source Class and Destination Class Usage

Configuring a Label Allocation and Substitution Policy for VPNs

You can control label-advertisements on MPLS ingress and AS border routers (ASBRs). Labels can be assigned on a per–next-hop (by default) or on a per-table basis (by configuring the *vrf-table-label* statement). This choice affects all routes of a given routing instance. You can also configure a policy to generate labels on a per-route basis by specifying a label allocation policy.

To specify a label allocation policy for the routing instance, configure the `label` statement and specify a label allocation policy using the **allocation** option:

```
label {
    allocation label-allocation-policy;
}
```

You can configure this statement at the following hierarchy levels:

- `[edit routing-instances routing-instance-name routing-options]`
- `[edit logical-systems logical-system-name routing-instances routing-instance-name routing-options]`

NOTE: The `[edit logical-systems]` hierarchy level is not applicable in ACX Series routers.

To configure the label allocation policy, include the `label-allocation` statement at the `[edit policy-options policy-statement policy-statement-name term term-name then]` hierarchy level. You can configure the label allocation mode as either **per-nexthop** or **per-table**.

For a VPN option B ASBR, labels for transit routes are substituted for a local virtual tunnel label or vrf-table-label label. When a VRF table is configured on the ASBR (this type of configuration is uncommon for the option B model), the ASBR does not generate MPLS swap or swap and push state for transit routes. Instead, the ASBR re-advertises a local virtual-tunnel or vrf-table-label label and forwards that transit traffic based on IP forwarding tables. The label substitution helps to conserve labels on Juniper Networks routers.

However, this type of label substitution effectively breaks the MPLS forwarding path, which becomes visible when using an MPLS OAM command such as LSP ping. You can configure the way in which labels are substituted on a per-route basis by specifying a label substitution policy.

To specify a label substitution policy for the routing instance, configure the `label` statement and specify a label substitution policy using the **substitution** option:

```
label {
    substitution label-substitution-policy;
}
```

You can configure this statement at the following hierarchy levels:

- `[edit routing-instances routing-instance-name routing-options]`
- `[edit logical-systems logical-system-name routing-instances routing-instance-name routing-options]`

NOTE: The `[edit logical-systems]` hierarchy level is not applicable in ACX Series routers.

The label substitution policy is used to determine whether or not a label should be substituted on an ASBR router. The results of the policy operation are either **accept** (label substitution is performed) or **reject** (label substitution is not performed). The default behavior is **accept**. The following set command example illustrates how you can configure a **reject** label substitution policy: `set policy-options policy-statement no-label-substitution term default then reject`.

Distributing VPN Routes

IN THIS SECTION

This topic describes configuring a router to handle route information in BGP, MPLS signaling, and policies.

Enabling Routing Information Exchange for VPNs

For Layer 2 VPNs, Layer 3 VPNs, virtual-router routing instances, VPLS, EVPNs, and Layer 2 circuits to function properly, the service provider's PE and P routers must be able to exchange routing information. For this to happen, you must configure either an IGP (such as OSPF or IS-IS) or static routes on these routers. You configure the IGP on the master instance of the routing protocol process at the `[edit protocols]` hierarchy level, not within the routing instance used for the VPN—that is, not at the `[edit routing-instances]` hierarchy level.

When you configure the PE router, do not configure any summarization of the PE router's loopback addresses at the area boundary. Each PE router's loopback address should appear as a separate route.

Configuring IBGP Sessions Between PE Routers in VPNs

You must configure an IBGP session between the PE routers to allow the PE routers to exchange information about routes originating and terminating in the VPN. The PE routers rely on this information to determine which labels to use for traffic destined for remote sites.

Configure an IBGP session for the VPN as follows:

```
[edit protocols]
bgp {
    group group-name {
        type internal;
        local-address ip-address;
        family evpn {
            signaling;
        }
        family (inet-vpn | inet6-vpn) {
            unicast;
        }
        family l2vpn {
            signaling;
        }
        neighbor ip-address;
    }
}
```

The IP address in the `local-address` statement is the address of the loopback interface on the local PE router. The IBGP session for the VPN runs through the loopback address. (You must also configure the loopback interface at the `[edit interfaces]` hierarchy level.)

The IP address in the `neighbor` statement is the loopback address of the neighboring PE router. If you are using RSVP signaling, this IP address is the same address you specify in the `to` statement at the `[edit mpls label-switched-path lsp-path-name]` hierarchy level when you configure the MPLS LSP.

The `family` statement allows you to configure the IBGP session for Layer 2 VPNs, VPLS, EVPNs or for Layer 3 VPNs.

- To configure an IBGP session for Layer 2 VPNs and VPLS, include the `signaling` statement at the `[edit protocols bgp group group-name family l2vpn]` hierarchy level:

  ```
  [edit protocols bgp group group-name family l2vpn]
  signaling;
  ```

- To configure an IBGP session for EVPNs, include the `signaling` statement at the `[edit protocols bgp group group-name family evpn]` hierarchy level:

```
[edit protocols bgp group group-name family evpn]
signaling;
```

- To configure an IPv4 IBGP session for Layer 3 VPNs, configure the `unicast` statement at the `[edit protocols bgp group group-name family inet-vpn]` hierarchy level:

```
[edit protocols bgp group group-name family inet-vpn]
unicast;
```

- To configure an IPv6 IBGP session for Layer 3 VPNs, configure the `unicast` statement at the `[edit protocols bgp group group-name family inet6-vpn]` hierarchy level:

```
[edit protocols bgp group group-name family inet6-vpn]
unicast;
```

NOTE: You can configure both `family inet` and `family inet-vpn` or both `family inet6` and `family inet6-vpn` within the same peer group. This allows you to enable support for both IPv4 and IPv4 VPN routes or both IPv6 and IPv6 VPN routes within the same peer group.

Configuring Aggregate Labels for VPNs

Aggregate labels for VPNs allow a Juniper Networks routing platform to aggregate a set of incoming labels (labels received from a peer router) into a single forwarding label that is selected from the set of incoming labels. The single forwarding label corresponds to a single next hop for that set of labels. Label aggregation reduces the number of VPN labels that the router must examine.

For a set of labels to share an aggregate forwarding label, they must belong to the same forwarding equivalence class (FEC). The labeled packets must have the same destination egress interface.

Including the `community community-name` statement with the `aggregate-label` statement lets you specify prefixes with a common origin community. Set by policy on the peer PE, these prefixes represent an FEC on the peer PE router.

⚠ **CAUTION**: If the target community is set by mistake instead of the origin community, forwarding problems at the egress PE can result. All prefixes from the peer PE will appear to be in the same FEC, resulting in a single inner label for all CE routers behind a given PE in the same VPN.

To work with route reflectors in Layer 3 VPN networks, the Juniper Networks M10i router aggregates a set of incoming labels only when the routes:

- Are received from the same peer router
- Have the same site of origin community
- Have the same next hop

The next hop requirement is important because route reflectors forward routes originated from different BGP peers to another BGP peer without changing the next hop of those routes.

To configure aggregate labels for VPNs, include the *aggregate-label* statement:

```
aggregate-label {
    community community-name;
}
```

For a list of hierarchy levels at which you can include this statement, see the statement summary for this statement.

For information about how to configure a community, see Understanding BGP Communities, Extended Communities, and Large Communities as Routing Policy Match Conditions.

Configuring a Signaling Protocol and LSPs for VPNs

IN THIS SECTION

For VPNs to function, you must enable a signaling protocol, either the LDP or RSVP on the provider edge (PE) routers and on the provider (P) routers. You also need to configure label-switched paths (LSPs) between the ingress and egress routers. In a typical VPN configuration, you need to configure LSPs from each PE router to all of the other PE routers participating in the VPN in a full mesh.

> **NOTE**: As with any configuration involving MPLS, you cannot configure any of the core-facing interfaces on the PE routers over dense Fast Ethernet PICs.

To enable a signaling protocol, perform the steps in one of the following sections:

Using LDP for VPN Signaling

To use LDP for VPN signaling, perform the following steps on the PE and provider (P) routers:

1. Configure LDP on the interfaces in the core of the service provider's network by including the `ldp` statement at the `[edit protocols]` hierarchy level.

   You need to configure LDP only on the interfaces between PE routers or between PE and P routers. You can think of these as the "core-facing" interfaces. You do not need to configure LDP on the interface between the PE and customer edge (CE) routers.

   ```
   [edit]
   protocols {
       ldp {
           interface type-fpc/pic/port;
       }
   }
   ```

2. Configure the MPLS address family on the interfaces on which you enabled LDP (the interfaces you configured in Step "1" on page 69) by including the `family mpls` statement at the `[edit interfaces` *`type-fpc/pic/port`* `unit` *`logical-unit-number`*`]` hierarchy level.

   ```
   [edit]
   interfaces {
       type-fpc/pic/port {
           unit logical-unit-number {
               family mpls;
           }
       }
   }
   ```

3. Configure OSPF or IS-IS on each PE and P router.

You configure these protocols at the master instance of the routing protocol, not within the routing instance used for the VPN.

- To configure OSPF, include the `ospf` statement at the `[edit protocols]` hierarchy level. At a minimum, you must configure a backbone area on at least one of the router's interfaces.

  ```
  [edit]
  protocols {
      ospf {
          area 0.0.0.0 {
              interface type-fpc/pic/port;
          }
      }
  }
  ```

- To configure IS-IS, include the `isis` statement at the `[edit protocols]` hierarchy level and configure the loopback interface and International Organization for Standardization (ISO) family at the `[edit interfaces]` hierarchy level. At a minimum, you must enable IS-IS on the router, configure a network entity title (NET) on one of the router's interfaces (preferably the loopback interface, lo0), and configure the ISO family on all interfaces on which you want IS-IS to run. When you enable IS-IS, Level 1 and Level 2 are enabled by default. The following is the minimum IS-IS configuration. In the `address` statement, *address* is the NET.

  ```
  [edit]
  interfaces {
      lo0 {
          unit logical-unit-number {
              family iso {
                  address address;
              }
          }
      }
      type-fpc/pic/port {
          unit logical-unit-number {
              family iso;
          }
      }
  }
  protocols {
      isis {
          interface all;
  ```

```
    }
}
```

Using RSVP for VPN Signaling

To use RSVP for VPN signaling, perform the following steps:

1. On each PE router, configure traffic engineering.

   To do this, you must configure an interior gateway protocol (IGP) that supports traffic engineering (either IS-IS or OSPF) and enable traffic engineering support for that protocol.

   To enable OSPF traffic engineering support, include the `traffic-engineering` statement at the `[edit protocols ospf]` hierarchy level:

   ```
   [edit protocols ospf]
   traffic-engineering {
       shortcuts;
   }
   ```

   For IS-IS, traffic engineering support is enabled by default.

2. On each PE and P router, enable RSVP on the interfaces that participate in the label-switched path (LSP).

   On the PE router, these interfaces are the ingress and egress points to the LSP. On the P router, these interfaces connect the LSP between the PE routers. Do not enable RSVP on the interface between the PE and the CE routers, because this interface is not part of the LSP.

   To configure RSVP on the PE and P routers, include the `interface` statement at the `[edit protocols rsvp]` hierarchy level. Include one `interface` statement for each interface on which you are enabling RSVP.

   ```
   [edit protocols]
   rsvp {
       interface interface-name;
       interface interface-name;
   }
   ```

3. On each PE router, configure an MPLS LSP to the PE router that is the LSP’s egress point.

To do this, include the `interface` and `label-switched-path` statements at the `[edit protocols mpls]` hierarchy level:

```
[edit protocols]
mpls {
    interface interface-name;
    label-switched-path path-name {
        to ip-address;
    }
}
```

In the `to` statement, specify the address of the LSP's egress point, which is an address on the remote PE router.

In the `interface` statement, specify the name of the interface (both the physical and logical portions). Include one `interface` statement for the interface associated with the LSP.

When you configure the logical portion of the same interface at the `[edit interfaces]` hierarchy level, you must also configure the `family inet` and `family mpls` statements:

```
[edit interfaces]
interface-name {
    unit logical-unit-number {
        family inet;
        family mpls;
    }
}
```

4. On all P routers that participate in the LSP, enable MPLS by including the `interface` statement at the `[edit mpls]` hierarchy level.

   Include one `interface` statement for each connection to the LSP.

```
[edit]
mpls {
    interface interface-name;
    interface interface-name;
}
```

5. Enable MPLS on the interface between the PE and CE routers by including the `interface` statement at the `[edit mpls]` hierarchy level.

Doing this allows the PE router to assign an MPLS label to traffic entering the LSP or to remove the label from traffic exiting the LSP.

```
[edit]
mpls {
    interface interface-name;
}
```

For information about configuring MPLS, see the Configuring the Ingress Router for MPLS-Signaled LSPs.

SEE ALSO

Configuring the Ingress Router for MPLS-Signaled LSPs

Configuring Policies for the VRF Table on PE Routers in VPNs

IN THIS SECTION

On each PE router, you must define policies that define how routes are imported into and exported from the router's VRF table. In these policies, you must define the route target, and you can optionally define the route origin.

To configure policy for the VRF tables, you perform the steps in the following sections:

Configuring the Route Target

As part of the policy configuration for the VPN routing table, you must define a route target, which defines which VPN the route is a part of. When you configure different types of VPN services (Layer 2

VPNs, Layer 3 VPNs, EVPNs, or VPLS) on the same PE router, be sure to assign unique route target values to avoid the possibility of adding route and signaling information to the wrong VPN routing table.

To configure the route target, include the `target` option in the `community` statement:

```
community name members target:community-id;
```

You can include this statement at the following hierarchy levels:

- `[edit policy-options]`
- `[edit logical-systems logical-system-name policy-options]`

name is the name of the community.

community-id is the identifier of the community. Specify it in one of the following formats:

- *as-number:number*, where *as-number* is an AS number (a 2-byte value) and *number* is a 4-byte community value. The AS number can be in the range 1 through 65,535. We recommend that you use an IANA-assigned, nonprivate AS number, preferably the ISP's own or the customer's own AS number. The community value can be a number in the range 0 through 4,294,967,295 (2^{32} – 1).
- *ip-address:number*, where *ip-address* is an IPv4 address (a 4-byte value) and *number* is a 2-byte community value. The IP address can be any globally unique unicast address. We recommend that you use the address that you configure in the `router-id` statement, which is a nonprivate address in your assigned prefix range. The community value can be a number in the range 1 through 65,535.

Configuring the Route Origin

In the import and export policies for the PE router's VRF table, you can optionally assign the route origin (also known as the site of origin) for a PE router's VRF routes using a VRF export policy applied to multiprotocol external BGP (MP-EBGP) VPN IPv4 route updates sent to other PE routers.

Matching on the assigned route origin attribute in a receiving PE's VRF import policy helps ensure that VPN-IPv4 routes learned through MP-EBGP updates from one PE are not reimported to the same VPN site from a different PE connected to the same site.

To configure a route origin, complete the following steps:

1. Include the `community` statement with the `origin` option:

   ```
   community name members origin:community-id;
   ```

   You can include this statement at the following hierarchy levels:

- `[edit policy-options]`
- `[edit logical-systems` *`logical-system-name`* `policy-options]`

`name` is the name of the community.

`community-id` is the identifier of the community. Specify it in one of the following formats:

- *`as-number`*:*`number`*, where *`as-number`* is an AS number (a 2-byte value) and *`number`* is a 4-byte community value. The AS number can be in the range 1 through 65,535. We recommend that you use an IANA-assigned, nonprivate AS number, preferably the ISP's own or the customer's own AS number. The community value can be a number in the range 0 through 4,294,967,295 (2^{32} – 1).
- *`ip-address`*:*`number`*, where *`ip-address`* is an IPv4 address (a 4-byte value) and *`number`* is a 2-byte community value. The IP address can be any globally unique unicast address. We recommend that you use the address that you configure in the `router-id` statement, which is a nonprivate address in your assigned prefix range. The community value can be a number in the range 1 through 65,535.

2. Include the community in the import policy for the PE router's VRF table by configuring the `community` statement with the *`community-id`* identifier defined in Step "1" on page 74 at the `[edit policy-options policy-statement` *`import-policy-name`* `term` *`import-term-name`* `from]` hierarchy level. See "Configuring an Import Policy for the PE Router's VRF Table" on page 75.

   If the policy's `from` clause does not specify a community condition, the `vrf-import` statement in which the policy is applied cannot be committed. The Junos OS commit operation does not pass the validation check.

3. Include the community in the export policy for the PE router's VRF table by configuring the `community` statement with the *`community-id`* identifier defined in Step "1" on page 74 at the `[edit policy-options policy-statement` *`export-policy-name`* `term` *`export-term-name`* `then]` hierarchy level. See "Configuring an Export Policy for the PE Router's VRF Table" on page 77.

See "Configuring the Route Origin for VPNs" on page 81 for a configuration example.

Configuring an Import Policy for the PE Router's VRF Table

Each VPN can have a policy that defines how routes are imported into the PE router's VRF table. An import policy is applied to routes received from other PE routers in the VPN. A policy must evaluate all routes received over the IBGP session with the peer PE router. If the routes match the conditions, the route is installed in the PE router's *`routing-instance-name`*`.inet.0` VRF table. An import policy must contain a second term that rejects all other routes.

Unless an import policy contains only a `then reject` statement, it must include a reference to a community. Otherwise, when you try to commit the configuration, the commit fails. You can configure multiple import policies.

An import policy determines what to import to a specified VRF table based on the VPN routes learned from the remote PE routers through IBGP. The IBGP session is configured at the `[edit protocols bgp]` hierarchy level. If you also configure an import policy at the `[edit protocols bgp]` hierarchy level, the import policies at the `[edit policy-options]` hierarchy level and the `[edit protocols bgp]` hierarchy level are combined through a logical AND operation. This allows you to filter traffic as a group.

To configure an import policy for the PE router's VRF table, follow these steps:

1. To define an import policy, include the `policy-statement` statement. For all PE routers, an import policy must always include the `policy-statement` statement, at a minimum:

   ```
   policy-statement import-policy-name {
       term import-term-name {
           from {
               protocol bgp;
               community community-id;
           }
           then accept;
       }
       term term-name {
           then reject;
       }
   }
   ```

   You can include the `policy-statement` statement at the following hierarchy levels:

   - `[edit policy-options]`
   - `[edit logical-systems logical-system-name policy-options]`

   The *import-policy-name* policy evaluates all routes received over the IBGP session with the other PE router. If the routes match the conditions in the `from` statement, the route is installed in the PE router's *routing-instance-name*.inet.0 VRF table. The second term in the policy rejects all other routes.

   For more information about creating policies, see the Routing Policies, Firewall Filters, and Traffic Policers User Guide.

2. You can optionally use a regular expression to define a set of communities to be used for the VRF import policy.

For example you could configure the following using the `community` statement at the `[edit policy-options policy-statement` *`policy-statement-name`*`]` hierarchy level:

```
[edit policy-options vrf-import-policy-sample]
community high-priority members *:50
```

Note that you cannot configure a regular expression as a part of a route target extended community. For more information about how to configure regular expressions for communities, see Understanding How to Define BGP Communities and Extended Communities .

3. To configure an import policy, include the `vrf-import` statement:

```
vrf-import import-policy-name;
```

You can include this statement at the following hierarchy levels:

- `[edit routing-instances` *`routing-instance-name`*`]`
- `[edit logical-systems` *`logical-system-name`* `routing-instances` *`routing-instance-name`*`]`

Configuring an Export Policy for the PE Router's VRF Table

Each VPN can have a policy that defines how routes are exported from the PE router's VRF table. An export policy is applied to routes sent to other PE routers in the VPN. An export policy must evaluate all routes received over the routing protocol session with the CE router. (This session can use the BGP, OSPF, or Routing Information Protocol [RIP] routing protocols, or static routes.) If the routes match the conditions, the specified community target (which is the route target) is added to them and they are exported to the remote PE routers. An export policy must contain a second term that rejects all other routes.

Export policies defined within the VPN routing instance are the only export policies that apply to the VRF table. Any export policy that you define on the IBGP session between the PE routers has no effect on the VRF table. You can configure multiple export policies.

To configure an export policy for the PE router's VRF table, follow these steps:

1. For all PE routers, an export policy must distribute VPN routes to and from the connected CE routers in accordance with the type of routing protocol that you configure between the CE and PE routers within the routing instance.

To define an export policy, include the `policy-statement` statement. An export policy must always include the `policy-statement` statement, at a minimum:

```
policy-statement export-policy-name {
    term export-term-name {
        from protocol (bgp | ospf | rip | static);
        then {
            community add community-id;
            accept;
        }
    }
    term term-name {
        then reject;
    }
}
```

NOTE: Configuring the `community add` statement is a requirement for Layer 2 VPN VRF export policies. If you change the `community add` statement to the `community set` statement, the router at the egress of the Layer 2 VPN link might drop the connection.

NOTE: When configuring draft-rosen multicast VPNs operating in source-specific mode and using the `vrf-export` statement to specify the export policy, the policy must have a term that accepts routes from the vrf-name.mdt.0 routing table. This term ensures proper PE autodiscovery using the `inet-mdt` address family.

When configuring draft-rosen multicast VPNs operating in source-specific mode and using the `vrf-target` statement, the VRF export policy is automatically generated and automatically accepts routes from the vrf-name.mdt.0 routing table.

You can include the `policy-statement` statement at the following hierarchy levels:

- `[edit policy-options]`
- `[edit logical-systems logical-system-name policy-options]`

The *`export-policy-name`* policy evaluates all routes received over the routing protocol session with the CE router. (This session can use the BGP, OSPF, or RIP routing protocols, or static routes.) If the routes match the conditions in the `from` statement, the community target specified in the `then community add` statement is added to them and they are exported to the remote PE routers. The second term in the policy rejects all other routes.

For more information about creating policies, see the Routing Policies, Firewall Filters, and Traffic Policers User Guide.

2. To apply the policy, include the `vrf-export` statement:

```
vrf-export export-policy-name;
```

You can include this statement at the following hierarchy levels:

- `[edit routing-instances routing-instance-name]`
- `[edit logical-systems logical-system-name routing-instances routing-instance-name]`

Applying Both the VRF Export and the BGP Export Policies

When you apply a VRF export policy as described in "Configuring an Export Policy for the PE Router's VRF Table" on page 77, routes from VPN routing instances are advertised to other PE routers based on this policy, whereas the BGP export policy is ignored.

If you include the `vpn-apply-export` statement in the BGP configuration, both the VRF export and BGP group or neighbor export policies are applied (VRF first, then BGP) before routes are advertised in the VPN routing tables to other PE routers.

NOTE: When a PE device is also acting as a Route Reflector (RR) or an Autonomous system boundary router (ASBR) in a Carrier-over-Carrier or inter-AS VPN, the next-hop manipulation in the vrf-export policy is ignored.

When you include the `vpn-apply-export` statement, be aware of the following:

- Routes imported into the bgp.l3vpn.0 routing table retain the attributes of the original routes (for example, an OSPF route remains an OSPF route even when it is stored in the bgp.l3vpn.0 routing table). You should be aware of this when you configure an export policy for connections between an IBGP PE router and a PE router, a route reflector and a PE router, or AS boundary router (ASBR) peer routers.
- By default, all routes in the bgp.l3vpn.0 routing table are exported to the IBGP peers. If the last statement of the export policy is deny all and if the export policy does not specifically match on routes in the bgp.l3vpn.0 routing table, no routes are exported.

To apply both the VRF export and BGP export policies to VPN routes, include the `vpn-apply-export` statement:

```
vpn-apply-export;
```

For a list of hierarchy levels at which you can include this statement, see the statement summary section for this statement.

Configuring a VRF Target

Including the `vrf-target` statement in the configuration for a VRF target community causes default VRF import and export policies to be generated that accept and tag routes with the specified target community. You can still create more complex policies by explicitly configuring VRF import and export policies. These policies override the default policies generated when you configure the `vrf-target` statement.

If you do not configure the `import` and `export` options of the `vrf-target` statement, the specified community string is applied in both directions. The `import` and `export` keywords give you more flexibility, allowing you to specify a different community for each direction.

The syntax for the VRF target community is not a name. You must specify it in the format `target:`*x*`:`*y*. A community name cannot be specified because this would also require you to configure the community members for that community using the `policy-options` statement. If you define the `policy-options` statements, then you can just configure VRF import and export policies as usual. The purpose of the `vrf-target` statement is to simplify the configuration by allowing you to configure most statements at the `[edit routing-instances]` hierarchy level.

To configure a VRF target, include the `vrf-target` statement:

```
vrf-target community;
```

You can include this statement at the following hierarchy levels:

- `[edit routing-instances` *routing-instance-name*`]`
- `[edit logical-systems` *logical-system-name* `routing-instances` *routing-instance-name*`]`

An example of how you might configure the `vrf-target` statement follows:

```
[edit routing-instances sample]
vrf-target target:69:102;
```

To configure the `vrf-target` statement with the `export` and `import` options, include the following statements:

```
vrf-target {
    export community-name;
    import community-name;
}
```

You can include this statement at the following hierarchy levels:

- `[edit routing-instances routing-instance-name]`
- `[edit logical-systems logical-system-name routing-instances routing-instance-name]`

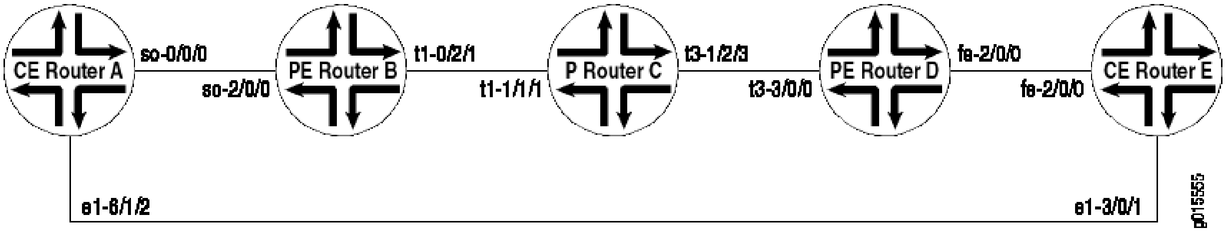
Configuring the Route Origin for VPNs

IN THIS SECTION

You can use route origin to prevent routes learned from one customer edge (CE) router marked with origin community from being advertised back to it from another CE router in the same AS.

In the example, the route origin is used to prevent routes learned from CE Router A that are marked with origin community from being advertised back to CE Router E by AS 200. The example topology is shown in Figure 9 on page 82.

Figure 9: Network Topology of Site of Origin Example



In this topology, CE Router A and CE Router E are in the same AS (AS200). They use EBGP to exchange routes with their respective provider edge (PE) routers, PE Router B and PE Router D. The two CE routers have a back connection.

The following sections describe how to configure the route origin for a group of VPNs:

Configuring the Site of Origin Community on CE Router A

The following section describes how to configure CE Router A to advertise routes with a site of origin community to PE Router B for this example.

NOTE: In this example, direct routes are configured to be advertised, but any route can be configured.

Configure a policy to advertise routes with `my-soo` community on CE Router A as follows:

```
[edit]
policy-options {
    policy-statement export-to-my-isp {
        term a {
            from {
                protocol direct;
            }
            then {
                community add my-soo;
                accept;
            }
        }
    }
}
```

Configuring the Community on CE Router A

Configure the `my-soo` community on CE Router A as follows:

```
[edit]
policy-options {
    community my-soo {
        members origin:100:1;
    }
}
```

Applying the Policy Statement on CE Router A

Apply the export-to-my-isp policy statement as an export policy to the EBGP peering on the CE Router A as follows:

```
[edit]
protocols {
    bgp {
        group my_isp {
            export export-to-my-isp;
        }
    }
}
```

When you issue the `show route receive-protocol bgp detail` command, you should see the following routes originated from PE Router B with `my-soo` community:

```
user@host> show route receive-protocol bgp 10.12.99.2 detail
inet.0: 16 destinations, 16 routes (15 active, 0 holddown, 1 hidden)
inet.3: 2 destinations, 2 routes (2 active, 0 holddown, 0 hidden)
vpn_blue.inet.0: 8 destinations, 10 routes (8 active, 0 holddown, 0 hidden)
* 10.12.33.0/30 (2 entries, 1 announced)
     Nexthop: 10.12.99.2
     AS path: 100 I
     Communities: origin:100:1
  10.12.99.0/30 (2 entries, 1 announced)
     Nexthop: 10.12.99.2
     AS path: 100 I
     Communities: origin:100:1
```

```
* 10.255.71.177/32 (1 entry, 1 announced)
     Nexthop: 10.12.99.2
     AS path: 100 I
     Communities: origin:100:1
* 192.168.64.0/21 (1 entry, 1 announced)
     Nexthop: 10.12.99.2
     AS path: 100 I
     Communities: origin:100:1
iso.0: 1 destinations, 1 routes (1 active, 0 holddown, 0 hidden)
mpls.0: 8 destinations, 8 routes (8 active, 0 holddown, 0 hidden)
bgp.l3vpn.0: 4 destinations, 4 routes (4 active, 0 holddown, 0 hidden)
inet6.0: 2 destinations, 2 routes (2 active, 0 holddown, 0 hidden)
__juniper_private1__.inet6.0: 1 destinations, 1 routes (1 active, 0 holddown, 0
hidden)
```

Configuring the Policy on PE Router D

Configure a policy on PE Router D that prevents routes with `my-soo` community tagged by CE Router A from being advertised to CE Router E as follows:

```
[edit]
policy-options {
    policy-statement soo-ce1-policy {
        term a {
            from {
                community my-soo;
                then {
                    reject;
                }
            }
        }
    }
}
```

Configuring the Community on PE Router D

Configure the community on PE Router D as follows:

```
[edit]
policy-options {
```

```
    community my-soo {
        members origin:100:1;
    }
}
```

Applying the Policy on PE Router D

To prevent routes learned from CE Router A from being advertised to CE Router E (the two routers can communicate these routes directly), apply the `soo-ce1-policy` policy statement as an export policy to the PE Router D and CE Router E EBGP session `vpn_blue`.

View the EBGP session on PE Router D using the `show routing-instances` command.

```
user@host# show routing-instances
vpn_blue {
    instance-type vrf;
    interface fe-2/0/0.0;
    vrf-target target:100:200;
    protocols {
        bgp {
            group ce2 {
                advertise-peer-as;
                peer-as 100;
                neighbor 10.12.99.6;
            }
        }
    }
}
```

Apply the `soo-ce1-policy` policy statement as an export policy to the PE Router D and CE Router E EBGP session `vpn_blue` as follows:

```
[edit routing-instances]
vpn_blue {
    protocols {
        bgp {
            group ce2{
                export soo-ce1-policy;
            }
        }
```

```
    }
}
```

RELATED DOCUMENTATION

Example: Configuring IS-IS

https://www.juniper.net/documentation/en_US/junos/information-products/pathway-pages/config-guide-routing/config-guide-ospf.html

Example: Configure a Basic MPLS-Based Layer 3 VPN | **229**

Route Target Filtering

IN THIS SECTION

This topic describes configuring static, BGP, and Proxy BGP route target filtering and provides examples on configuring route target filtering for VPNs.

Configuring Static Route Target Filtering for VPNs

The BGP VPN route target extended community (RFC 4360, *BGP Extended Communities Attribute*) is used to determine VPN membership. Static route target filtering helps to prevent resources from being consumed in portions of the network where the VPN routes are not needed due to the lack of member PE routers (RFC 4684, *Constrained Route Distribution for Border Gateway Protocol/MultiProtocol Label Switching (BGP/MPLS) Internet Protocol (IP) Virtual Private Networks (VPNs)*). Routers can originate routes into the RT-Constrain protocol to indicate their interest in receiving VPN routes containing route targets that match the RT-Constrain NLRI.

To configure static route target filtering for VPNs:

- Configure the `route-target-filter` statement at the `[edit routing-options rib bgp.rtarget.0 static]` hierarchy level.

  The following example illustrates how you could configure the *route-target-filter* statement:

  ```
  [edit routing-options rib bgp.rtarget.0 static]
  route-target-filter destination {
      group bgp-group;
      local;
      neighbor bgp-peer;
  }
  ```

- You can display route target filtering information using the `show bgp group rtf detail` command.

Reducing Network Resource Use with Static Route Target Filtering for VPNs

The BGP VPN route target extended community (RFC 4360, *BGP Extended Communities Attribute*) is used to determine VPN membership. Static route target filtering helps to prevent resources from being consumed in portions of the network where the VPN routes are not needed due to the lack of member PE routers (RFC 4684, *Constrained Route Distribution for Border Gateway Protocol/MultiProtocol Label Switching (BGP/MPLS) Internet Protocol (IP) Virtual Private Networks (VPNs)*). Routers can originate routes into the RT-Constrain protocol to indicate their interest in receiving VPN routes containing route targets that match the RT-Constrain NLRI.

Normally, for the RT-Constrain feature to function properly, it must be broadly deployed throughout a network. If this is not the case, the feature is less useful, because the RT-Constrain BGP speaker facing a non-RT-Constrain speaker must advertise a default RT-Constrain route to the other RT-Constrain speakers on behalf of the peer that does not support the feature. This effectively removes the resource saving benefits of the feature in portions of the network where it is not supported since a default RT-Constrain route causes the PE router and all intervening PE routers to need to receive all VPN routes.

The static RT-Constrain feature enables you to partially deploy the RT-Constrain feature in a network. The feature is enabled at a boundary in the network where RT-Constrain is configured. However, some BGP VPN peers do not support RT-Constrain, typically PE routers. The route targets of those PE routers must be statically configured on the router. These route targets are disseminated using the RT-Constrain protocol.

The proxy RT-Constrain feature permits BGP VPN peers that do not support the protocol to have their route-targets discovered and disseminated automatically. However, this feature can only support symmetric route-targets. For example, the import and export route-targets for a VRF routing instance are identical. However, for a hub-and-spoke VPN, the import and export route-targets are not identical. In this scenario, the import and export route-target may be statically configured to be disseminated in the RT-Constrain protocol.

Configuring BGP Route Target Filtering for VPNs

IN THIS SECTION

BGP route target filtering allows you to distribute VPN routes to only the routers that need them. In VPN networks without BGP route target filtering configured, BGP distributes all VPN routes to all VPN peer routers.

For more information about BGP route target filtering, see RFC 4684, *Constrained Route Distribution for Border Gateway Protocol/MultiProtocol Label Switching (BGP/MPLS) Internet Protocol (IP) Virtual Private Networks (VPNs)*.

The following sections provide an overview of BGP route target filtering and how to configure it for VPNs:

BGP Route Target Filtering Overview

PE routers, unless they are configured as route reflectors or are running an EBGP session, discard any VPN routes that do not include a route target extended community as specified in the local VRF import policies. This is the default behavior of the Junos OS.

However, unless it is explicitly configured not to store VPN routes, any router configured either as a route reflector or border router for a VPN address family must store all of the VPN routes that exist in the service provider’s network. Also, though PE routers can automatically discard routes that do not include a route target extended community, route updates continue to be generated and received.

By reducing the number of routers receiving VPN routes and route updates, BGP route target filtering helps to limit the amount of overhead associated with running a VPN. BGP route target filtering is most effective at reducing VPN-related administrative traffic in networks where there are many route reflectors or AS border routers that do not participate in the VPNs directly (not acting as PE routers for the CE devices).

BGP route target filtering uses standard UPDATE messages to distributes route target extended communities between routers. The use of UPDATE messages allows BGP to use its standard loop detection mechanisms, path selection, policy support, and database exchange implementation.

Configuring BGP Route Target Filtering for VPNs

BGP route target filtering is enabled through the exchange of the `route-target` address family, stored in the bgp.rtarget.0 routing table. Based on the `route-target` address family, the route target NLRI (address family indicator [AFI]=1, subsequent AFI [SAFI]=132) is negotiated with its peers.

On a system that has locally configured VRF instances, BGP automatically generates local routes corresponding to targets referenced in the `vrf-import` policies.

To configure BGP route target filtering, include the `family route-target` statement:

```
family route-target {
    advertise-default;
    external-paths number;
    prefix-limit number;
}
```

For a list of hierarchy levels at which you can include this statement, see the statement summary section for this statement.

The `advertise-default`, `external-paths`, and `prefix-limit` statements affect the BGP route target filtering configuration as follows:

- The `advertise-default` statement causes the router to advertise the default route target route (0:0:0/0) and suppress all routes that are more specific. This can be used by a route reflector on BGP groups consisting of neighbors that act as PE routers only. PE routers often need to advertise all routes to the route reflector.

  Suppressing all route target advertisements other than the default route reduces the amount of information exchanged between the route reflector and the PE routers. The Junos OS further helps to reduce route target advertisement overhead by not maintaining dependency information unless a nondefault route is received.

- The `external-paths` statement (which has a default value of 1) causes the router to advertise the VPN routes that reference a given route target. The number you specify determines the number of external peer routers (currently advertising that route target) that receive the VPN routes.

- The `prefix-limit` statement limits the number of prefixes that can be received from a peer router.

The `route-target`, `advertise-default`, and `external-path` statements affect the RIB-OUT state and must be consistent between peer routers that share the same BGP group. The `prefix-limit` statement affects the receive side only and can have different settings between different peer routers in a BGP group.

SEE ALSO

Configuring the Route Origin for VPNs | **81**

Example: BGP Route Target Filtering for VPNs

BGP route target filtering is enabled by configuring the `family route-target` statement at the appropriate BGP hierarchy level. This statement enables the exchange of a new `route-target` address family, which is stored in the bgp.rtarget.0 routing table.

The following configuration illustrates how you could configure BGP route target filtering for a BGP group titled `to_vpn04`:

```
[edit]
protocols {
    bgp {
        group to_vpn04 {
            type internal;
            local-address 10.255.14.182;
            peer-as 200;
            neighbor 10.255.14.174 {
                family inet-vpn {
                    unicast;
                }
                family route-target;
            }
        }
    }
}
```

The following configuration illustrates how you could configure a couple of local VPN routing and forwarding (VRF) routing instances to take advantage of the functionality provided by BGP route target filtering. Based on this configuration, BGP would automatically generate local routes corresponding to the route targets referenced in the VRF import policies (note the targets defined by the `vrf-target` statements).

```
[edit]
routing-instances {
    vpn1 {
        instance-type vrf;
        interface t1-0/1/2.0;
        vrf-target target:200:101;
        protocols {
            ospf {
                export bgp-routes;
                area 0.0.0.0 {
                    interface t1-0/1/2.0;
                }
            }
        }
    }
    vpn2 {
        instance-type vrf;
        interface t1-0/1/2.1;
        vrf-target target:200:102;
        protocols {
            ospf {
                export bgp-routes;
                area 0.0.0.0 {
                    interface t1-0/1/2.1;
                }
            }
        }
    }
}
```

Issue the **show route table bgp.rtarget.0** show command to verify the BGP route target filtering configuration:

```
user@host> show route table bgp.rtarget.0
 bgp.rtarget.0: 4 destinations, 6 routes (4 active, 0 holddown, 0 hidden)
 + = Active Route, - = Last Active, * = Both
200:200:101/96
                    *[RTarget/5] 00:10:00
```

```
                       Local
200:200:102/96
                    *[RTarget/5] 00:10:00
                       Local
200:200:103/96
                    *[BGP/170] 00:09:48, localpref 100, from 10.255.14.174
                       AS path: I
                     > t3-0/0/0.0
200:200:104/96
                    *[BGP/170] 00:09:48, localpref 100, from 10.255.14.174
                       AS path: I
                     > t3-0/0/0.0
```

The `show` command display format for route target prefixes is:

```
AS number:route target extended community/length
```

The first number represents the autonomous system (AS) of the router that sent this advertisement. The remainder of the display follows the Junos `show` command convention for extended communities.

The output from the `show route table bgp-rtarget.0` command displays the locally generated and remotely generated routes.

The first two entries correspond to the route targets configured for the two local VRF routing instances (`vpn1` and `vpn2`):

- `200:200:101/96`—Community `200:101` in the `vpn1` routing instance
- `200:200:102/96`—Community `200:102` in the `vpn2` routing instance

The last two entries are prefixes received from a BGP peer:

- `200:200:103/96`—Tells the local router that routes tagged with this community (`200:103`) should be advertised to peer `10.255.14.174` through `t3-0/0/0.0`
- `200:200:104/96`—Tells the local router that routes tagged with this community (`200:104`) should be advertised to peer `10.255.14.174` through `t3-0/0/0.0`

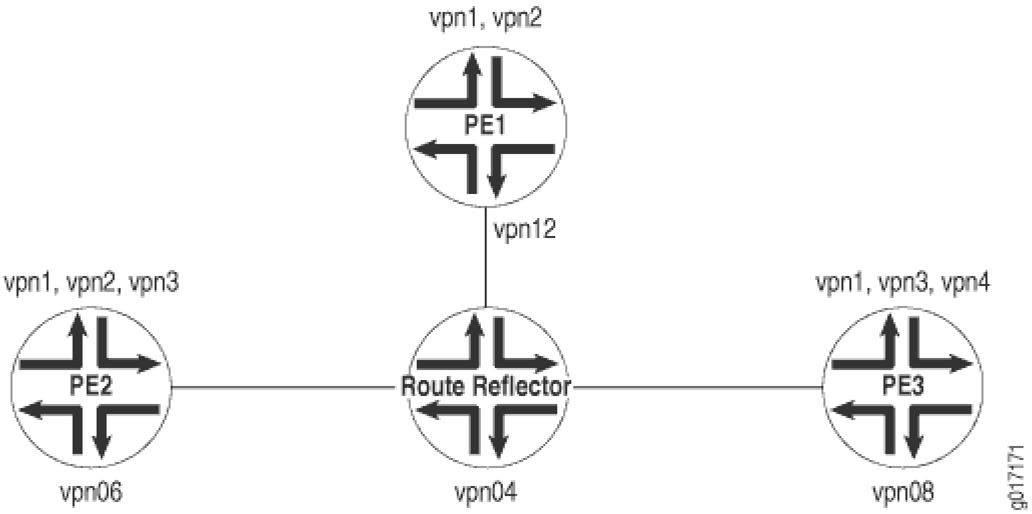
Example: Configuring BGP Route Target Filtering for VPNs

IN THIS SECTION

BGP route target filtering reduces the number of routers that receive VPN routes and route updates, helping to limit the amount of overhead associated with running a VPN. BGP route target filtering is most effective at reducing VPN-related administrative traffic in networks where there are many route reflectors or AS border routers that do not participate in the VPNs directly (do not act as PE routers for the CE devices).

Figure 10 on page 93 illustrates the topology for a network configured with BGP route target filtering for a group of VPNs.

Figure 10: BGP Route Target Filtering Enabled for a Group of VPNs



The following sections describe how to configure BGP route target filtering for a group of VPNs:

Configure BGP Route Target Filtering on Router PE1

This section describes how to enable BGP route target filtering on Router PE1 for this example.

Configure the routing options on router PE1 as follows:

```
[edit]
routing-options {
    route-distinguisher-id 10.255.14.182;
    autonomous-system 198;
}
```

Configure the BGP protocol on Router PE1 as follows:

```
[edit]
protocols {
    bgp {
        group to_VPN_D {
            type internal;
            local-address 10.255.14.182;
            peer-as 198;
            neighbor 10.255.14.174 {
                family inet-vpn {
                    unicast;
                }
                family route-target;
            }
        }
    }
}
```

Configure the `vpn1` routing instance as follows:

```
[edit]
routing-instances {
    vpn1 {
        instance-type vrf;
        interface t1-0/1/2.0;
        vrf-target target:198:101;
        protocols {
            ospf {
                export bgp-routes;
                area 0.0.0.0 {
                    interface t1-0/1/2.0;
                }
```

```
            }
        }
    }
}
```

Configure the `vpn2` routing instance on Router PE1 as follows:

```
[edit]
routing-instances {
    vpn2 {
        instance-type vrf;
        interface t1-0/1/2.1;
        vrf-target target:198:102;
        protocols {
            ospf {
                export bgp-routes;
                area 0.0.0.0 {
                    interface t1-0/1/2.1;
                }
            }
        }
    }
}
```

Once you have implemented this configuration, you should see the following when you issue a `show route table bgp.rtarget.0` command:

```
user@host> show route table bgp.rtarget.0
 bgp.rtarget.0: 4 destinations, 6 routes (4 active, 0 holddown, 0 hidden)
 + = Active Route, - = Last Active, * = Both

 198.198:101/96
                    *[RTarget/5] 00:27:42
                       Local
                     [BGP/170] 00:27:30, localpref 100, from
 10.255.14.174
                       AS path: I
                     > via t3-0/0/0.0
 198.198:102/96
                    *[RTarget/5] 00:27:42
                       Local
```

```
                       [BGP/170] 00:27:30, localpref 100, from
10.255.14.174
                         AS path: I
                       > via t3-0/0/0.0
198.198:103/96
                      *[BGP/170] 00:27:30, localpref 100, from
10.255.14.174
                         AS path: I
                       > via t3-0/0/0.0
198.198:104/96
                      *[BGP/170] 00:27:30, localpref 100, from
10.255.14.174
                         AS path: I
                       > via t3-0/0/0.0
```

Configure BGP Route Target Filtering on Router PE2

This section describes how to enable BGP route target filtering on Router PE2 for this example.

Configure the routing options on Router PE2 as follows:

```
[edit]
routing-options {
    route-distinguisher-id 10.255.14.176;
    autonomous-system 198;
}
```

Configure the BGP protocol on Router PE2 as follows:

```
[edit]
protocols {
    bgp {
        group to_vpn04 {
            type internal;
            local-address 10.255.14.176;
            peer-as 198;
            neighbor 10.255.14.174 {
                family inet-vpn {
                    unicast;
                }
                family route-target;
```

```
                }
            }
        }
    }
```

Configure the `vpn1` routing instance on Router PE2 as follows:

```
[edit]
routing-instances {
    vpn1 {
        instance-type vrf;
        interface t3-0/0/0.0;
        vrf-target target:198:101;
        protocols {
            bgp {
                group vpn1 {
                    type external;
                    peer-as 101;
                    as-override;
                    neighbor 10.49.11.2;
                }
            }
        }
    }
}
```

Configure the `vpn2` routing instance on Router PE2 as follows:

```
[edit]
routing-instances {
    vpn2 {
        instance-type vrf;
        interface t3-0/0/0.1;
        vrf-target target:198:102;
        protocols {
            bgp {
                group vpn2 {
                    type external;
                    peer-as 102;
                    as-override;
                    neighbor 10.49.21.2;
```

```
                }
            }
        }
    }
}
```

Configure the `vpn3` routing instance on Router PE2 as follows:

```
[edit]
routing-instances {
    vpn3 {
        instance-type vrf;
        interface t3-0/0/0.2;
        vrf-import vpn3-import;
        vrf-export vpn3-export;
        protocols {
            bgp {
                group vpn3 {
                    type external;
                    peer-as 103;
                    as-override;
                    neighbor 10.49.31.2;
                }
            }
        }
    }
}
```

Once you have configured router PE2 in this manner, you should see the following when you issue the `show route table bgp.rtarget.0` command:

```
user@host> show route table bgp.rtarget.0
 bgp.rtarget.0: 4 destinations, 7 routes (4 active, 0 holddown, 0 hidden)
 + = Active Route, - = Last Active, * = Both

 198.198:101/96
                   *[RTarget/5] 00:28:15
                      Local
                    [BGP/170] 00:28:03, localpref 100, from
 10.255.14.174
                      AS path: I
```

```
                    > via t1-0/1/0.0
198.198:102/96
                   *[RTarget/5] 00:28:15
                      Local
                    [BGP/170] 00:28:03, localpref 100, from
10.255.14.174
                      AS path: I
                    > via t1-0/1/0.0
198.198:103/96
                   *[RTarget/5] 00:28:15
                      Local
                    [BGP/170] 00:28:03, localpref 100, from
10.255.14.174
                      AS path: I
                    > via t1-0/1/0.0
198.198:104/96
                   *[BGP/170] 00:28:03, localpref 100, from
10.255.14.174
                      AS path: I
                    > via t1-0/1/0.0
```

Configure BGP Route Target Filtering on the Route Reflector

This section illustrates how to enable BGP route target filtering on the route reflector for this example.

Configure the routing options on the route reflector as follows:

```
[edit]
routing-options {
    route-distinguisher-id 10.255.14.174;
    autonomous-system 198;
}
```

Configure the BGP protocol on the route reflector as follows:

```
[edit]
protocols {
    bgp {
        group rr-group {
            type internal;
            local-address 10.255.14.174;
```

```
              cluster 10.255.14.174;
              peer-as 198;
              neighbor 10.255.14.182 {
                  description to_PE1_vpn12;
                  family inet-vpn {
                      unicast;
                  }
                  family route-target;
              }
              neighbor 10.255.14.176 {
                  description to_PE2_vpn06;
                  family inet-vpn {
                      unicast;
                  }
                  family route-target;
              }
              neighbor 10.255.14.178 {
                  description to_PE3_vpn08;
                  family inet-vpn {
                      unicast;
                  }
                  family route-target;
              }
          }
      }
  }
```

Once you have configured the route reflector in this manner, you should see the following when you issue the `show route table bgp.rtarget.0` command:

```
user@host> show route table bgp.rtarget.0
 bgp.rtarget.0: 4 destinations, 8 routes (4 active, 0 holddown, 0 hidden)
 + = Active Route, - = Last Active, * = Both

 198.198:101/96
                    *[BGP/170] 00:29:03, localpref 100, from
 10.255.14.176
                       AS path: I
                     > via t1-0/2/0.0
                     [BGP/170] 00:29:03, localpref 100, from
 10.255.14.178
                       AS path: I
```

```
                       > via t3-0/1/1.0
                       [BGP/170] 00:29:03, localpref 100, from
10.255.14.182
                         AS path: I
                       > via t3-0/1/3.0
198.198:102/96
                      *[BGP/170] 00:29:03, localpref 100, from
10.255.14.176
                         AS path: I
                       > via t1-0/2/0.0
                       [BGP/170] 00:29:03, localpref 100, from
10.255.14.182
                         AS path: I
                       > via t3-0/1/3.0
198.198:103/96
                      *[BGP/170] 00:29:03, localpref 100, from
10.255.14.176
                         AS path: I
                       > via t1-0/2/0.0
                       [BGP/170] 00:29:03, localpref 100, from
10.255.14.178
                         AS path: I
                       > via t3-0/1/1.0
198.198:104/96
                      *[BGP/170] 00:29:03, localpref 100, from
10.255.14.178
                         AS path: I
                       > via t3-0/1/1.0
```

Configure BGP Route Target Filtering on Router PE3

The following section describes how to enable BGP route target filtering on Router PE3 for this example.

Configure the routing options on Router PE3 as follows:

```
[edit]
routing-options {
    route-distinguisher-id 10.255.14.178;
    autonomous-system 198;
}
```

Configure the BGP protocol on Router PE3 as follows:

```
[edit]
protocols {
    bgp {
        group to_vpn04 {
            type internal;
            local-address 10.255.14.178;
            peer-as 198;
            neighbor 10.255.14.174 {
                family inet-vpn {
                    unicast;
                }
                family route-target;
            }
        }
    }
}
```

Configure the `vpn1` routing instance on Router PE3 as follows:

```
[edit]
routing-instances {
    vpn1 {
        instance-type vrf;
        interface t3-0/0/0.0;
        vrf-target target:198:101;
        protocols {
            rip {
                group vpn1 {
                    export bgp-routes;
                    neighbor t3-0/0/0.0;
                }
            }
        }
    }
}
```

Configure the `vpn3` routing instance on Router PE3 as follows:

```
[edit]
routing-instances {
    vpn3 {
        instance-type vrf;
        interface t3-0/0/0.1;
        vrf-target target:198:103;
        protocols {
            rip {
                group vpn3 {
                    export bgp-routes;
                    neighbor t3-0/0/0.1;
                }
            }
        }
    }
}
```

Configure the `vpn4` routing instance on Router PE3 as follows:

```
[edit]
routing-instances {
    vpn4 {
        instance-type vrf;
        interface t3-0/0/0.2;
        vrf-target target:198:104;
        protocols {
            rip {
                group vpn4 {
                    export bgp-routes;
                    neighbor t3-0/0/0.2;
                }
            }
        }
    }
}
```

Once you have configured Router PE3 in this manner, you should see the following when you issue the `show route table bgp.rtarget.0` command:

```
 user@host> show route table bgp.rtarget.0
bgp.rtarget.0: 4 destinations, 7 routes (4 active, 0 holddown, 0 hidden)
 + = Active Route, - = Last Active, * = Both

 198.198:101/96
                   *[RTarget/5] 00:29:42
                      Local
                    [BGP/170] 00:29:30, localpref 100, from
 10.255.14.174
                      AS path: I
                    > via t3-0/0/1.0
 198.198:102/96
                   *[BGP/170] 00:29:29, localpref 100, from
 10.255.14.174
                      AS path: I
                    > via t3-0/0/1.0
 198.198:103/96
                   *[RTarget/5] 00:29:42
                      Local
                    [BGP/170] 00:29:30, localpref 100, from
 10.255.14.174
                      AS path: I
                    > via t3-0/0/1.0
 198.198:104/96
                   *[RTarget/5] 00:29:42
                      Local
                    [BGP/170] 00:29:30, localpref 100, from
 10.255.14.174
                      AS path: I
                    > via t3-0/0/1.0
```

Example: Configuring an Export Policy for BGP Route Target Filtering for VPNs

IN THIS SECTION

This example shows how to configure an export routing policy for BGP route target filtering (also known as route target constrain, or RTC).

Requirements

This example uses the following hardware and software components:

- Four Juniper Networks devices that support BGP route target filtering.
- Junos OS Release 12.2 or later on one or more devices configured for proxy BGP route filtering. In this example, you explicitly configure proxy BGP route filtering on the route reflectors.

Before configuring an export policy for BGP route target filtering, make sure that you are familiar with and understand the following concepts:

- *Layer 2 VPNs*
- "Understanding Layer 3 VPNs" on page 6
- "Understanding VPN-IPv4 Addresses and Route Distinguishers" on page 224
- "Configuring Policies for the VRF Table on PE Routers in VPNs" on page 73
- "Configuring BGP Route Target Filtering for VPNs" on page 88
- BGP extended communities

Overview

IN THIS SECTION

BGP route target filtering allows you to reduce network resource consumption by distributing route target membership (RT membership) advertisements throughout the network. BGP uses the RT membership information to send VPN routes only to the devices that need them in the network. Similar to other types of BGP reachability, you can apply a routing policy to route target filtering routes to influence the network. When route target filtering is configured, restricting the flow of route target filtering routes also restricts the VPN routes that might be attracted by this RT membership. Configuring this policy involves:

- Creating a filter that defines the list of route target prefixes.
- Creating a policy to select a subset of the route target filters to use for BGP route target filtering.

To define the list of route target prefixes:

- You configure the `rtf-prefix-list` statement at the `[edit policy-options]` hierarchy level to specify the name of the route target prefix list and one or more route target prefixes to use. This configuration allows you to specify the incoming route target filtering routes that the device will use and then distribute them throughout the network.

To configure the routing policy and apply the route target prefix list to that policy, you can specify the following policy options:

- `family route-target`—(Optional) The route-target family match condition specifies matching BGP route target filtering routes. You define this criteria in the `from` statement. This example shows how to create an export policy using the `family route-target` match condition.

  NOTE: Juniper uses the inet.3 table to resolve the next hop address when `family route-target` is configured.

- `protocol route-target`—(Optional) The route-target protocol match condition defines the criteria that an incoming route must match. You define this criteria in the `from` statement. This statement is primarily useful for restricting the policy to locally generated route target filtering routes.

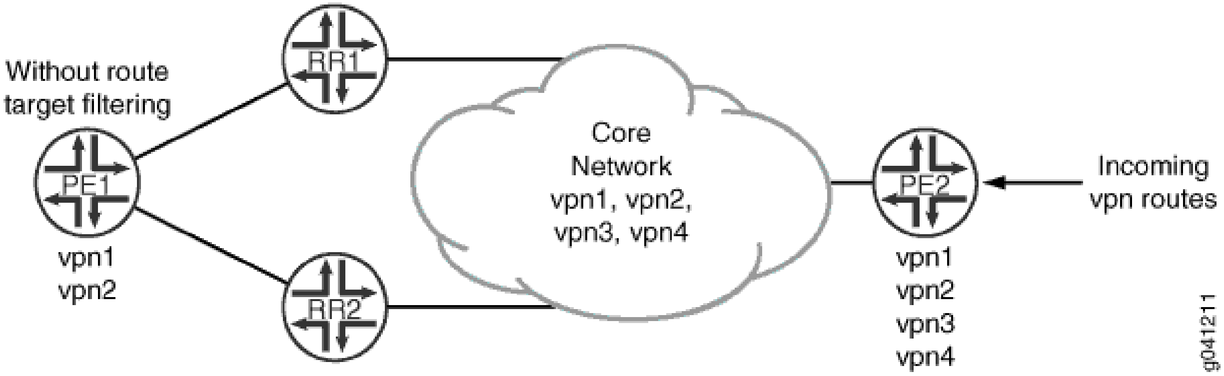
  NOTE: When you use the `show route table bgp.rtarget.0` command to view proxy BGP route target filtering routes, you will see the BGP protocol for received routes and the route target protocol routes for local route target filtering routes.

- `rtf-prefix-list` *`name`*—The rtf-prefix-list statement applies the list of route target prefixes that you already configured to the policy. You define this criteria in the `from` statement.

Topology Diagram

Figure 11 on page 107 shows the topology used in this example.

Figure 11: BGP Route Target Filtering Export Policy Topology



In this example, BGP route target filtering is configured on the route reflectors (Device RR1 and Device RR2) and provider edge (PE) Device PE2. The other PE, Device PE1, does not support BGP route target filtering. Proxy BGP route target filtering is also configured on the peering sessions between the route reflectors and Device PE1 to minimize the number of VPN route updates processed by Device PE1. Device PE2 has four VPNs configured (vpn1, vpn2, vpn3, and vpn4), and Device PE1 has two VPNs configured (vpn1 and vpn2). In the sample topology, all devices participate in autonomous system (AS) 203, OSPF is the configured interior gateway protocol (IGP), and LDP is the signaling protocol used by the VPNs. In this example, we use static routes in the VPN routing and forwarding (VRF) instances to generate VPN routes. This is done in place of using a PE to customer edge (CE) protocol such as OSPF or BGP.

In this example, you further control the routes being advertised from Device PE2 to Device PE1 by configuring an export policy on Device PE2 to prevent vpn3 routes from being advertised to Device RR1. You create a policy that specifies the `family route-target` match condition, defines the list of route target prefixes, and applies the list of route target prefixes by defining the `rtf-prefix-list` criteria.

Configuration

IN THIS SECTION

CLI Quick Configuration

To quickly configure this example, copy the following commands, paste them into a text file, remove any line breaks, change any details necessary to match your network configuration, and then copy and paste the commands into the CLI at the `[edit]` hierarchy level.

Device PE1

```
set interfaces ge-1/0/0 unit 0 description PE1-to-RR1
set interfaces ge-1/0/0 unit 0 family inet address 10.49.0.1/30
set interfaces ge-1/0/0 unit 0 family mpls
set interfaces ge-1/0/1 unit 0 description PE1-to-RR2
set interfaces ge-1/0/1 unit 0 family inet address 10.49.10.1/30
set interfaces ge-1/0/1 unit 0 family mpls
set protocols ldp interface ge-1/0/0
set protocols ldp interface ge-1/0/1
set protocols bgp group internal type internal
set protocols bgp group internal local-address 10.255.163.58
set protocols bgp group internal neighbor 10.255.165.220 family inet-vpn unicast
set protocols bgp group internal neighbor 10.255.165.28 family inet-vpn unicast
set protocols ospf area 0.0.0.0 interface ge-1/0/0
set protocols ospf area 0.0.0.0 interface ge-1/0/1
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set routing-options route-distinguisher-id 10.255.163.58
set routing-options autonomous-system 203
set routing-instances vpn1 instance-type vrf
set routing-instances vpn1 vrf-target target:203:100
set routing-instances vpn1 routing-options static route 203.0.113.1/24 discard
set routing-instances vpn2 instance-type vrf
set routing-instances vpn2 vrf-target target:203:101
set routing-instances vpn2 routing-options static route 203.0.113.2/24 discard
```

Device RR1

```
set interfaces ge-1/0/0 unit 0 description RR1-to-PE1
set interfaces ge-1/0/0 unit 0 family inet address 10.49.0.2/30
set interfaces ge-1/0/0 unit 0 family mpls
set interfaces ge-1/0/1 unit 0 description RR1-to-PE2
set interfaces ge-1/0/1 unit 0 family inet address 10.50.0.2/30
set interfaces ge-1/0/1 unit 0 family mpls
set protocols ldp interface ge-1/0/0
set protocols ldp interface ge-1/0/1
```

```
set protocols bgp group internal type internal
set protocols bgp group internal local-address 198.51.100.0
set protocols bgp group internal cluster 198.51.100.1
set protocols bgp group internal neighbor 10.255.163.58 description vpn1-to-pe1 family inet-vpn
unicast
set protocols bgp group internal neighbor 10.255.163.58 family route-target proxy-generate
set protocols bgp group internal neighbor 10.255.168.42 description vpn1-to-pe2 family inet-vpn
unicast
set protocols ospf area 0.0.0.0 interface ge-1/0/0
set protocols ospf area 0.0.0.0 interface ge-1/0/1
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set routing-options route-distinguisher-id 10.255.165.220
set routing-options autonomous-system 203
```

Device RR2

```
set interfaces ge-1/0/0 unit 0 description RR2-to-PE1
set interfaces ge-1/0/0 unit 0 family inet address 10.49.10.2/30
set interfaces ge-1/0/0 unit 0 family mpls
set interfaces ge-1/0/1 unit 0 description RR2-to-PE2
set interfaces ge-1/0/1 unit 0 family inet address 10.50.10.2/30
set interfaces ge-1/0/1 unit 0 family mpls
set protocols ldp interface ge-1/0/0
set protocols ldp interface ge-1/0/1
set protocols bgp group internal type internal
set protocols bgp group internal local-address 10.255.165.28
set protocols bgp group internal cluster 198.51.100.1
set protocols bgp group internal neighbor 10.255.163.58 description vpn2-to-pe1 family inet-vpn
unicast
set protocols bgp group internal neighbor 10.255.163.58 family route-target proxy-generate
set protocols bgp group internal neighbor 10.255.168.42 description vpn2-to-pe2 family inet-vpn
unicast
set protocols bgp group internal neighbor 10.255.163.58 family route-target
set protocols ospf area 0.0.0.0 interface ge-1/0/0
set protocols ospf area 0.0.0.0 interface ge-1/0/1
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set routing-options route-distinguisher-id 10.255.165.28
set routing-options autonomous-system 203
```

Device PE2

```
set interfaces ge-1/0/0 unit 0 description PE2-to-RR1
set interfaces ge-1/0/0 unit 0 family inet address 10.50.0.1/30
set interfaces ge-1/0/0 unit 0 family mpls
set interfaces ge-1/0/1 unit 0 description PE2-to-RR2
set interfaces ge-1/0/1 unit 0 family inet address 10.50.10.2/30
set interfaces ge-1/0/1 unit 0 family mpls
set protocols ldp interface ge-1/0/0
set protocols ldp interface ge-1/0/1
set protocols bgp group internal type internal
set protocols bgp group internal local-address 10.255.168.42
set protocols bgp group internal family inet-vpn unicast
set protocols bgp group internal family route-target
set protocols bgp group internal neighbor 10.255.165.220 export filter-rtc
set protocols bgp group internal neighbor 10.255.165.28
set protocols ospf area 0.0.0.0 interface ge-1/0/0
set protocols ospf area 0.0.0.0 interface ge-1/0/1
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set policy-options rtf-prefix-list exclude-103 203:203:103/96
set policy-options policy-statement filter-rtc from family route-target
set policy-options policy-statement filter-rtc from rtf-prefix-list exclude-103
set policy-options policy-statement filter-rtc then reject
set routing-options route-distinguisher-id 10.255.168.42
set routing-options autonomous-system 203
set routing-instances vpn1 instance-type vrf
set routing-instances vpn1 vrf-target target:203:100
set routing-instances vpn1 routing-options static route 203.0.113.1/24 discard
set routing-instances vpn2 instance-type vrf
set routing-instances vpn2 vrf-target target:203:101
set routing-instances vpn2 routing-options static route 203.0.113.2/24 discard
set routing-instances vpn3 instance-type vrf
set routing-instances vpn3 vrf-target target:203:103
set routing-instances vpn3 routing-options static route 203.0.113.3/24 discard
set routing-instances vpn4 instance-type vrf
set routing-instances vpn4 vrf-target target:203:104
set routing-instances vpn4 routing-options static route 203.0.113.4/24 discard
```

Configuring Device PE1

Step-by-Step Procedure

The following example requires you to navigate various levels in the configuration hierarchy. For information about navigating the CLI, see *Using the CLI Editor in Configuration Mode*.

To configure Device PE1:

1. Configure the interfaces.

   ```
   [edit interfaces]
   user@PE1# set ge-1/0/0 unit 0 description PE1-to-RR1
   user@PE1# set ge-1/0/0 unit 0 family inet address 10.49.0.1/30
   user@PE1# set ge-1/0/0 unit 0 family mpls
   user@PE1#set ge-1/0/1 unit 0 description PE1-to-RR2
   user@PE1#set ge-1/0/1 unit 0 family inet address 10.49.10.1/30
   user@PE1# set ge-1/0/1 unit 0 family mpls
   ```

2. Configure the route distinguisher and the AS number.

   ```
   [edit routing-options]
   user@PE1# set route-distinguisher-id 10.255.163.58
   user@PE1# set autonomous-system 203
   ```

3. Configure LDP as the signaling protocol used by the VPN.

   ```
   [edit protocols ldp]
   user@PE1# set interface ge-1/0/0
   user@PE1# set interface ge-1/0/1
   ```

4. Configure BGP.

   ```
   [edit protocols bgp group internal]
   user@PE1# set type internal
   user@PE1# set local-address 10.255.163.58
   user@PE1# set neighbor 10.255.165.220 family inet-vpn unicast
   user@PE1# set neighbor 10.255.165.28 family inet-vpn unicast
   ```

5. Configure OSPF.

   ```
   [edit protocols ospf area 0.0.0.0]
   user@PE1# set interface ge-1/0/0
   user@PE1# set interface ge-1/0/1
   user@PE1# set interface lo0.0 passive
   ```

6. Configure the VPN routing instances.

   ```
   [edit routing-instances vpn1]
   user@PE1# set instance-type vrf
   user@PE1# set vrf-target target:203:100
   user@PE1# set routing-options static route 203.0.113.1/24 discard
   ```

   ```
   [edit routing-instances vpn2]
   user@PE1# set instance-type vrf
   user@PE1# set vrf-target target:203:101
   user@PE1# set routing-options static route 203.0.113.2/24 discard
   ```

7. If you are done configuring the device, commit the configuration.

   ```
   [edit]
   user@PE1# commit
   ```

Results

From configuration mode, confirm your configuration by entering the `show interfaces`, `show protocols`, `show routing-options`, and `show routing-instances` commands. If the output does not display the intended configuration, repeat the instructions in this example to correct the configuration.

```
user@PE1# show interfaces
ge-1/0/0 {
    unit 0 {
        description PE1-to-RR1;
        family inet {
            address 10.49.0.1/30;
        }
```

```
        family mpls;
    }
}
ge-1/0/1 {
    unit 0 {
        description PE1-to-RR2;
        family inet {
            address 10.49.10.1/30;
        }
        family mpls;
    }
}
```

```
user@PE1# show protocols
bgp {
    group internal {
        type internal;
        local-address 10.255.163.58;
        neighbor 10.255.165.220 {
            family inet-vpn {
                unicast;
            }
        }
        neighbor 10.255.165.28 {
            family inet-vpn {
                unicast;
            }
        }
    }
}
ospf {
    area 0.0.0.0 {
        interface ge-1/0/0.0;
        interface ge-1/0/1.0;
        interface lo0.0 {
            passive;
        }
    }
}
ldp {
    interface ge-1/0/0.0;
```

```
    interface ge-1/0/1.0;
}
```

```
user@PE1# show routing-options
route-distinguisher-id 10.255.14.182;
autonomous-system 203;
```

```
user@PE1# show routing-instances
vpn1 {
    instance-type vrf;
    vrf-target target:203:100;
    routing-options {
        static {
            route 203.0.113.1/24 discard;
        }
    }
}
vpn2 {
    instance-type vrf;
    vrf-target target:203:101;
    routing-options {
        static {
            route 203.0.113.2/24 discard;
        }
    }
}
```

Configuring Device RR1

Step-by-Step Procedure

The following example requires you to navigate various levels in the configuration hierarchy. For information about navigating the CLI, see *Using the CLI Editor in Configuration Mode*.

To configure Device RR1:

1. Configure the interfaces.

```
[edit interfaces]
user@RR1# set ge-1/0/0 unit 0 description RR1-to-PE1
user@RR1# set ge-1/0/0 unit 0 family inet address 10.49.0.2/30
user@RR1# set ge-1/0/0 unit 0 family mpls
user@RR1# set ge-1/0/1 unit 0 description RR1-to-PE2
user@RR1# set ge-1/0/1 unit 0 family inet address 10.50.0.2/30
user@RR1# set ge-1/0/1 unit 0 family mpls
```

2. Configure the route distinguisher and the AS number.

```
[edit routing-options]
user@RR1# set route-distinguisher-id 10.255.165.220
user@RR1# set autonomous-system 203
```

3. Configure LDP as the signaling protocol used by the VPN.

```
[edit protocols ldp]
user@RR1# set interface ge-1/0/0
user@RR1# set interface ge-1/0/1
```

4. Configure BGP.

```
[edit protocols bgp group internal]
user@RR1# set type internal
user@RR1# set local-address 10.255.165.220
user@RR1# set cluster 198.51.100.1
user@RR1# set neighbor 10.255.163.58 description vpn1-to-pe1 family inet-vpn unicast
user@RR1# set neighbor 10.255.168.42 description vpn1-to-pe2 family inet-vpn unicast
```

5. Configure BGP route target filtering on the peering session with Device PE2.

```
[edit protocols bgp group internal]
user@RR1# set neighbor 10.255.168.42 family route-target
```

6. Configure proxy BGP route target filtering on the peering session with Device PE1.

```
[edit protocols bgp group internal]
user@RR1# set neighbor 10.255.163.58 family route-target proxy-generate
```

7. Configure OSPF.

```
[edit protocols ospf area 0.0.0.0]
user@RR1# set interface ge-1/0/0
user@RR1# set interface ge-1/0/1
user@RR1# set interface lo0.0 passive
```

8. If you are done configuring the device, commit the configuration.

```
[edit]
user@RR1# commit
```

Results

From configuration mode, confirm your configuration by entering the `show interfaces`, `show protocols`, and `show routing-options` commands. If the output does not display the intended configuration, repeat the instructions in this example to correct the configuration.

```
user@RR1# show interfaces
ge-1/0/0 {
    unit 0 {
        description RR1-to-PE1;
        family inet {
            address 10.49.0.2/30;
        }
        family mpls;
    }
}
ge-1/0/1 {
    unit 0 {
        description RR1-to-PE2;
        family inet {
            address 10.50.0.2/30;
```

```
        }
        family mpls;
    }
}
```

```
user@RR1# show protocols
bgp {
    group internal {
        type internal;
        local-address 198.51.100.0;
        cluster 198.51.100.1;
        neighbor 10.255.163.58 {
            description vpn1-to-pe1;
            family inet-vpn {
                unicast;
            }
            family route-target {
                proxy-generate;
            }
        }
        neighbor 10.255.168.42 {
            description vpn1-to-pe2;
            family inet-vpn {
                unicast;
            }
            family route-target;
        }
    }
}
ospf {
    area 0.0.0.0 {
        interface ge-1/0/0.0;
        interface ge-1/0/1.0;
        interface lo0.0 {
            passive;
        }
    }
}
ldp {
    interface ge-1/0/0.0;
    interface ge-1/0/1.0;
```

```
}
ospf {
    area 0.0.0.0 {
        interface ge-1/0/0.0;
        interface ge-1/0/1.0;
        interface lo0.0 {
            passive;
        }
    }
}
ldp {
    interface ge-1/0/0.0;
    interface ge-1/0/1.0;
}
```

```
user@RR1# show routing-options
route-distinguisher-id 10.255.165.220;
autonomous-system 203;
```

Configuring Device RR2

Step-by-Step Procedure

The following example requires you to navigate various levels in the configuration hierarchy. For information about navigating the CLI, see *Using the CLI Editor in Configuration Mode*.

To configure Device RR2:

1. Configure the interfaces.

   ```
   [edit interfaces]
   user@RR2# set ge-1/0/0 unit 0 description RR2-to-PE1
   user@RR2# set ge-1/0/0 unit 0 family inet address 10.49.10.2/30
   user@RR2# set ge-1/0/0 unit 0 family mpls
   user@RR2# set ge-1/0/1 unit 0 description RR2-to-PE2
   user@RR2# set ge-1/0/1 unit 0 family inet address 10.50.10.2/30
   user@RR2# set ge-1/0/1 unit 0 family mpls
   ```

2. Configure the route distinguisher and the AS number.

```
[edit routing-options]
user@RR2# set route-distinguisher-id 10.255.165.28
user@RR2# set autonomous-system 203
```

3. Configure LDP as the signaling protocol used by the VPN.

```
[edit protocols ldp]
user@RR2# set interface ge-1/0/0
user@RR2# set interface ge-1/0/1
```

4. Configure BGP.

```
[edit protocols bgp group internal]
user@RR2# set type internal
user@RR2# set local-address 10.255.165.28
user@RR2# set cluster 198.51.100.1
user@RR2# set neighbor 10.255.163.58 description vpn2-to-pe1 family inet-vpn unicast
user@RR2# set neighbor 10.255.168.42 description vpn2-to-pe2 family inet-vpn unicast
```

5. Configure BGP route target filtering on the peering session with Device PE2.

```
[edit protocols bgp group internal]
user@RR2# set neighbor 10.255.168.42 family route-target
```

6. Configure proxy BGP route target filtering on the peering session with Device PE1.

```
[edit protocols bgp group internal]
user@RR2# set neighbor 10.255.163.58 family route-target proxy-generate
```

7. Configure OSPF.

```
[edit protocols ospf area 0.0.0.0]
user@RR2# set interface ge-1/0/0
```

```
user@RR2# set interface ge-1/0/1
user@RR2# set interface lo0.0 passive
```

8. If you are done configuring the device, commit the configuration.

```
[edit]
user@RR2# commit
```

Results

From configuration mode, confirm your configuration by entering the `show interfaces`, `show protocols`, and `show routing-options` commands. If the output does not display the intended configuration, repeat the instructions in this example to correct the configuration.

```
user@RR2# show interfaces
ge-1/0/0 {
    unit 0 {
        description RR2-to-PE1;
        family inet {
            address 10.49.10.2/30;
        }
        family mpls;
    }
}
ge-1/0/1 {
    unit 0 {
        description RR2-to-PE2;
        family inet {
            address 10.50.10.2/30;
        }
        family mpls;
    }
}
```

```
user@RR2# show protocols
bgp {
    group internal {
        local-address 10.255.165.28;
        cluster 198.51.100.1;
```

```
        neighbor 10.255.163.58 {
            description vpn2-to-pe1;
            family inet-vpn {
                unicast;
            }
            family route-target {
                proxy-generate;
            }
        }
        neighbor 10.255.168.42 {
            description vpn2-to-pe2;
            family inet-vpn {
                unicast;
            }
            family route-target;
        }
    }
}
ospf {
    area 0.0.0.0 {
        interface ge-1/0/0.0;
        interface ge-1/0/1.0;
        interface lo0.0 {
            passive;
        }
    }
}
ldp {
    interface ge-1/0/0.0;
    interface ge-1/0/1.0;
}
```

```
user@RR2# show routing-options
route-distinguisher-id 10.255.165.28;
autonomous-system 203;
```

Configuring Device PE2

Step-by-Step Procedure

The following example requires you to navigate various levels in the configuration hierarchy. For information about navigating the CLI, see *Using the CLI Editor in Configuration Mode*.

To configure Device PE2:

1. Configure the interfaces.

   ```
   [edit interfaces]
   user@PE2# set ge-1/0/0 unit 0 description PE2-to-RR1
   user@PE2# set ge-1/0/0 unit 0 family inet address 10.50.0.1/30
   user@PE2# set ge-1/0/0 unit 0 family mpls
   user@PE2#set ge-1/0/1 unit 0 description PE2-to-RR2
   user@PE2#set ge-1/0/1 unit 0 family inet address 10.50.10.2/30
   user@PE2# set ge-1/0/1 unit 0 family mpls
   ```

2. Configure the route distinguisher and the AS number.

   ```
   [edit routing-options]
   user@PE2# set route-distinguisher-id 10.255.168.42
   user@PE2# set autonomous-system 203
   ```

3. Configure LDP as the signaling protocol used by the VPN.

   ```
   [edit protocols ldp]
   user@PE2# set interface ge-1/0/0
   user@PE2# set interface ge-1/0/1
   ```

4. Configure BGP.

   ```
   [edit protocols bgp group internal]
   user@PE2# set type internal
   user@PE2# set local-address 10.255.168.42
   user@PE2# set family inet-vpn unicast
   user@PE2# set family route-target
   ```

```
user@PE2# set neighbor 10.255.165.220
user@PE2# set neighbor 10.255.165.28
```

5. Configure OSPF.

```
[edit protocols ospf area 0.0.0.0]
user@PE2# set interface ge-1/0/0
user@PE2# set interface ge-1/0/1
user@PE2# set interface lo0.0 passive
```

6. Configure the VPN routing instances.

```
[edit routing-instances vpn1]
user@PE2# set instance-type vrf
user@PE2# set vrf-target target:203:100
user@PE2# set routing-options static route 203.0.113.1/24 discard
```

```
[edit routing-instances vpn2]
user@PE2# set instance-type vrf
user@PE2# set vrf-target target:203:101
user@PE2# set routing-options static route 203.0.113.2/24 discard
```

```
[edit routing-instances vpn3]
user@PE2# set instance-type vrf
user@PE2# set vrf-target target:203:103
user@PE2# set routing-options static route 203.0.113.3/24 discard
```

```
[edit routing-instances vpn4]
user@PE2# set instance-type vrf
user@PE2# set vrf-target target:203:104
user@PE2# set routing-options static route 203.0.113.4/24 discard
```

7. Configure and apply the export routing policy.

```
[edit policy-options]
user@PE2# set rtf-prefix-list exclude-103 203:203:103/96
```

```
[edit policy-options policy-statement filter-rtc]
user@PE2# set from family route-target
user@PE2# set from rtf-prefix-list exclude-103
user@PE2# set then reject
[edit protocols bgp group internal]
user@PE2# set neighbor 10.255.165.220 export filter-rtc
```

8. If you are done configuring the device, commit the configuration.

```
[edit]
user@PE2# commit
```

Results

From configuration mode, confirm your configuration by entering the `show interfaces`, `show protocols`, `show policy-options`, `show routing-options`, and `show routing-instances` commands. If the output does not display the intended configuration, repeat the instructions in this example to correct the configuration.

```
user@PE2# show interfaces
ge-1/0/0 {
    unit 0 {
        description PE2-to-RR1;
        family inet {
            address 10.50.0.1/30;
        }
        family mpls;
    }
}
ge-1/0/1 {
    unit 0 {
        description PE2-to-RR2;
        family inet {
            address 10.50.10.2/30;
        }
        family mpls;
```

```
    }
}
```

```
user@PE2# show protocols
    bgp {
        group internal {
            type internal;
            local-address 10.255.168.42;
            family inet-vpn {
                unicast;
            }
            family route-target;
            neighbor 10.255.165.220 {
                export filter-rtc;
            }
            neighbor 10.255.165.28;
        }
    }
ospf {
    area 0.0.0.0 {
        interface ge-1/0/0.0;
        interface ge-1/0/1.0;
        interface lo0.0 {
            passive;
        }
    }
}
ldp {
    interface ge-1/0/0.0;
    interface ge-1/0/1.0;
}
```

```
user@PE2# show routing-options
route-distinguisher-id 10.255.168.42;
autonomous-system 203;
```

```
user@PE2# show policy-options
policy-statement filter-rtc {
    from {
```

```
        family route-target;
        rtf-prefix-list exclude-103;
    }
    then reject;
}
rtf-prefix-list exclude-103 {
    203:203:103/96;
}
```

```
user@PE2# show routing-instances
vpn1 {
    instance-type vrf;
    vrf-target target:203:100;
    routing-options {
        static {
            route 203.0.113.1/24 discard;
        }
    }
}
vpn2 {
    instance-type vrf;
    vrf-target target:203:101;
    routing-options {
        static {
            route 203.0.113.2/24 discard;
        }
    }
}
vpn3 {
    instance-type vrf;
    vrf-target target:203:103;
    routing-options {
        static {
            route 203.0.113.3/24 discard;
        }
    }
}
vpn4 {
    instance-type vrf;
    vrf-target target:203:104;
    routing-options {
```

```
        static {
            route 203.0.113.4/24 discard;
        }
    }
}
```

Verification

IN THIS SECTION

Confirm that the configuration is working properly.

Verifying the Route Target Filtering Routes in the bgp.rtarget.0 Routing Table for Device RR1

Purpose

Verify that the route prefix for vpn3 is not in Device RR1's bgp.rtarget.0 table. Since an export policy on Device PE2 was applied to prevent the advertisement of vpn3 routes to Device RR1, Device RR1 should not receive those advertisements.

Action

From operational mode, enter the `show route advertising-protocol bgp 10.255.165.220 table bgp.rtarget.0` command.

```
user@PE2# show route advertising-protocol bgp 10.255.165.220 table bgp.rtarget.0
bgp.rtarget.0: 4 destinations, 11 routes
(4 active, 0 holddown, 0 hidden)
  Prefix                  Nexthop              MED     Lclpref    AS path
  203:203:100/96          *                    Self      100        I
  203:203:101/96          *                    Self      100        I
  203:203:104/96          *                    Self      100        I
```

Meaning

The bgp.rtartget.0 table does not display 203:203:103/96, which is the route prefix for vpn3. That means the export policy was applied correctly.

Verifying the Route Target Filtering Routes in the bgp.rtarget.0 Routing Table for Device RR2

Purpose

Verify that the route prefix for vpn3 is in Device RR2’s bgp.rtarget.0 table. Since an export policy was not applied on Device PE2 to prevent the advertisement of vpn3 routes to Device RR2, Device RR2 should receive advertisements from all of the VPNs.

Action

From operational mode, enter the `show route advertising-protocol bgp 10.255.165.28 table bgp.rtarget.0` command.

```
user@PE2# show route advertising-protocol bgp 10.255.165.28 table bgp.rtarget.0
bgp.rtarget.0: 4 destinations, 11 routes (4 active, 0 holddown, 0 hidden)
(4 active, 0 holddown, 0 hidden)
  Prefix                    Nexthop                 MED     Lclpref     AS path
  203:203:100/96            *                       Self       100        I
  203:203:101/96            *                       Self       100        I
  203:203:103/96            *                       Self       100        I
  203:203:104/96            *                       Self       100        I
```

Meaning

The bgp.rtartget.0 table displays the route prefixes for all of the VPNs.

Example: Configuring Layer 3 VPN Protocol Family Qualifiers for Route Filters

IN THIS SECTION

This example shows how to control the scope of BGP import policies by configuring a family qualifier for the BGP import policy. The family qualifier specifies routes of type `inet`, `inet6`, `inet-vpn`, or `inet6-vpn`.

Requirements

This example uses Junos OS Release 10.0 or later.

Before you begin:

- Configure the device interfaces.
- Configure an interior gateway protocol. See the Junos OS Routing Protocols Library.
- Configure a BGP session for multiple route types. For example, configure the session for both family `inet` routes and family `inet-vpn` routes. See "Configuring IBGP Sessions Between PE Routers in VPNs" on page 66 and "Configuring Layer 3 VPNs to Carry IPv6 Traffic" on page 285.

Overview

Family qualifiers cause a route filter to match only one specific family. When you configure an IPv4 route filter without a family qualifier, as shown here, the route filter matches `inet` and `inet-vpn` routes.

```
route-filter ipv4-address/mask;
```

Likewise, when you configure an IPv6 route filter without a family qualifier, as shown here, the route filter matches `inet6` and `inet6-vpn` routes.

```
route-filter ipv6-address/mask;
```

Consider the case in which a BGP session has been configured for both family `inet` routes and family `inet-vpn` routes, and an import policy has been configured for this BGP session. This means that both family `inet` and family `inet-vpn` routes, when received, share the same import policy. The policy term might look as follows:

```
from {
    route-filter 0.0.0.0/0 exact;
```

```
}
then {
    next-hop self;
    accept;
}
```

This route-filter logic matches an `inet` route of 0.0.0.0 and an `inet-vpn` route whose IPv4 address portion is 0.0.0.0. The 8-byte route distinguisher portion of the `inet-vpn` route is not considered in the route-filter matching. This is a change in Junos OS behavior that was introduced in Junos OS Release 10.0.

If you do not want your policy to match both types of routes, add a family qualifier to your policy. To have the route-filter match only `inet` routes, add the family `inet` policy qualifier. To have the route-filter match only `inet-vpn` routes, add the family `inet-vpn` policy qualifier.

The family qualifier is evaluated before the route-filter is evaluated. Thus, the route-filter is not evaluated if the family match fails. The same logic applies to family `inet6` and family `inet6-vpn`. The route-filter used in the `inet6` example must use an IPv6 address. There is a potential efficiency gain in using a family qualifier because the family qualifier is tested before most other qualifiers, quickly eliminating routes from undesired families.

Configuration

IN THIS SECTION

Procedure

CLI Quick Configuration

To quickly configure this example, copy the following commands, paste them into a text file, remove any line breaks, change any details necessary to match your network configuration, and then copy and paste the commands into the CLI at the `[edit]` hierarchy level.

inet Example

```
set policy-options policy-statement specific-family from family inet
set policy-options policy-statement specific-family from route-filter 0.0.0.0/0 exact
```

```
set policy-options policy-statement specific-family then next-hop self
set policy-options policy-statement specific-family then accept
set protocols bgp import specific-family
```

Inet-vpn Example

```
set policy-options policy-statement specific-family from family inet-vpn
set policy-options policy-statement specific-family from route-filter 0.0.0.0/0 exact
set policy-options policy-statement specific-family then next-hop self
set policy-options policy-statement specific-family then accept
set protocols bgp import specific-family
```

inet6 Example

```
set policy-options policy-statement specific-family from family inet6
set policy-options policy-statement specific-family from route-filter 0::0/0 exact
set policy-options policy-statement specific-family then next-hop self
set policy-options policy-statement specific-family then accept
set protocols bgp import specific-family
```

Inet6-vpn Example

```
set policy-options policy-statement specific-family from family inet6-vpn
set policy-options policy-statement specific-family from route-filter 0::0/0 exact
set policy-options policy-statement specific-family then next-hop self
set policy-options policy-statement specific-family then accept
set protocols bgp import specific-family
```

Step-by-Step Procedure

The following example requires that you navigate various levels in the configuration hierarchy. For information about navigating the CLI, see *Using the CLI Editor in Configuration Mode* in the CLI User Guide.

To configure a flow map:

1. Configure the family qualifier.

```
[edit policy-options]
user@host# set policy-statement specific-family from family inet
```

2. Configure the route filter.

```
[edit policy-options]
user@host# set policy-statement specific-family from route-filter 0.0.0.0/0 exact
```

3. Configure the policy actions.

```
[edit policy-options]
user@host# set policy-statement specific-family then next-hop self
user@host# set policy-statement specific-family then accept
```

4. Apply the policy.

```
[edit protocols bgp]
user@host# set import specific-family
```

Results

From configuration mode, confirm your configuration by issuing the `show protocols` and `show policy-options` command. If the output does not display the intended configuration, repeat the instructions in this example to correct the configuration.

```
user@host# show protocols
bgp {
    import specific-family;
}
user@host# show policy-options
policy-statement specific-family {
    from {
        family inet;
        route-filter 0.0.0.0/0 exact;
    }
    then {
```

```
            next-hop self;
            accept;
        }
    }
```

If you are done configuring the device, enter **commit** from configuration mode.

Repeat the procedure for every protocol family for which you need a specific route-filter policy.

Verification

To verify the configuration, run the following commands:

- `show route advertising-protocol bgp` *`neighbor`* `detail`
- `show route instance` *`instance-name`* `detail`

Understanding Proxy BGP Route Target Filtering for VPNs

BGP route target filtering (also known as route target constrain, or RTC) allows you to distribute VPN routes to only the devices that need them. In VPN networks without BGP route target filtering configured, BGP distributes all VPN routes to all VPN peer devices, which can strain network resources. The route target filtering feature was introduced to reduce the number of devices receiving VPN routes and VPN routing updates, thereby limiting the amount of overhead associated with running a VPN. The Junos OS implementation for BGP route target filtering is based on RFC 4684, *Constrained Route Distribution for Border Gateway Protocol/MultiProtocol Label Switching (BGP/MPLS) Internet Protocol (IP) Virtual Private Networks (VPNs)*.

What if you have a network environment where route target filtering is not widely deployed, or what if some devices do not support route target filtering? For example, you might have a BGP speaker with route target filtering enabled that is peered with a BGP speaker that does not support or have route target filtering configured. In this case, the BGP speaker with route target filtering configured must advertise default route target membership (RT membership) on behalf of its peer. The route target filtering resource savings are unrealized because the device supporting the filtering must now send all VPN routes to the device that does not support the filter. Proxy BGP route target filtering (or Proxy RTC) permits the generation of RT membership for devices that do not support route target filtering. This eases the deployment of route target filtering in networks where it is incompletely deployed or not fully supported.

Proxy BGP route target filtering allows you to distribute proxy RT membership advertisements created from the received BGP VPN routes to other devices in the network that need them. These are known as proxy advertisements because the device creates the RT membership on behalf of its peers without the route target filtering functionality. Proxy BGP route target filtering uses BGP route target extended communities that are exported to a specific BGP speaker to generate the route targets. Generated proxy RTC routes are stored in the bgp.rtarget.0 routing table.

You can also configure a policy to control which VPN routes are used to generate the proxy RTC routes. This can help control which RT membership is generated by the proxying device. In addition, you can configure a policy to reduce the memory overhead associated with proxy RTC. Proxy RTC only uses additional memory on a per-VPN route basis when it is permitted by a policy to be used for generating RT membership.

Example: Configuring Proxy BGP Route Target Filtering for VPNs

IN THIS SECTION

This example shows how to configure proxy BGP route target filtering (also known as proxy route target constrain, or proxy RTC).

Requirements

This example uses the following hardware and software components:

- Four Juniper Networks devices that can be a combination of M Series, MX Series, or T Series routers.
- Junos OS Release 12.2 or later on one or more devices configured for proxy BGP route filtering. In this example, you explicitly configure proxy BGP route filtering on the route reflectors.

Before configuring proxy BGP route target filtering, make sure that you are familiar with and understand the following concepts:

- *Layer 2 VPNs*
- "Understanding Layer 3 VPNs" on page 6
- "Understanding VPN-IPv4 Addresses and Route Distinguishers" on page 224
- "Configuring Policies for the VRF Table on PE Routers in VPNs" on page 73
- "Configuring BGP Route Target Filtering for VPNs" on page 88
- BGP extended communities

Overview

IN THIS SECTION

Route target filtering decreases the number of devices in a network that receive VPN routes that are not needed. Proxy BGP route target filtering allows networks to take advantage of route target filtering in locations where the feature is not currently supported. By configuring this feature, you can realize many of the same network resource savings that are available to you if your network fully supported BGP route target filtering.

To configure proxy BGP route target filtering, you include the `family route-target proxy-generate` statement on the devices that will distribute proxy route target membership (RT membership) advertisements for the devices that do not support BGP route target filtering. The proxy BGP route target filtering routes are then stored in the bgp.rtarget.0 routing table.

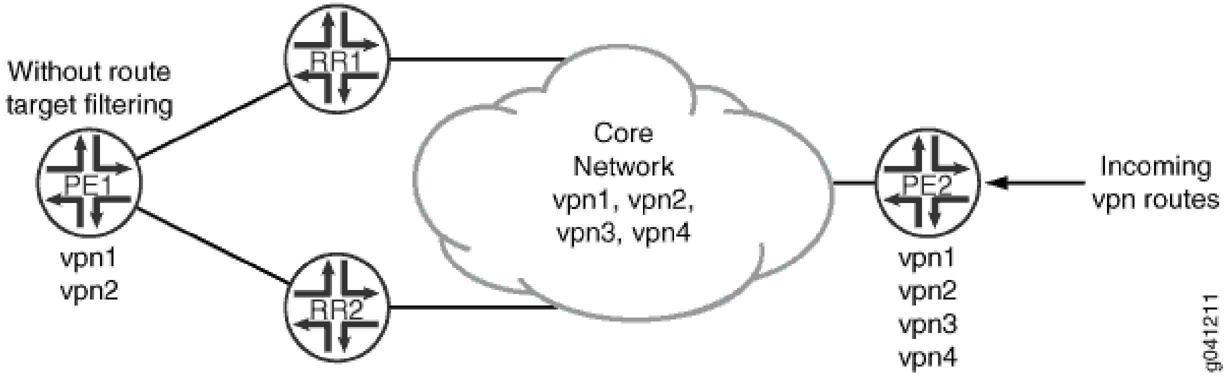
Proxy BGP route target filtering is intended to create RT membership advertisements for devices that do not support the BGP route target filtering feature. If the `proxy-generate` statement is present, but the route target family is negotiated with the BGP peer, the proxy-generate functionality is disabled. This allows simplified configuration of BGP peer groups where a portion of the peers in the group support route target filtering but others do not. In such an example case, the `family route-target proxy-generate` statement might be part of the BGP peer group configuration.

NOTE: When deploying proxy BGP route target filtering in your network, the `advertise-default` statement for BGP route target filtering causes the device to advertise the default route target route (0:0:0/0) and suppress all routes that are more specific. If you have proxy BGP route target filtering configured on one device and one or more peers have the `advertise-default` statement configured as part of their BGP route target filtering configuration, the advertise-default configuration is ignored.

Topology Diagram

Figure 12 on page 136 shows the topology used in this example.

Figure 12: Proxy BGP Route Target Filtering Topology



In this example, BGP route target filtering is configured on the route reflectors (Device RR1 and Device RR2) and the provider edge (PE) Device PE2, but the other PE, Device PE1, does not support the BGP route target filtering functionality. Device PE2 has four VPNs configured (vpn1, vpn2, vpn3, and vpn4). Device PE1 has two VPNs configured (vpn1 and vpn2), so this device is only interested in receiving route updates for vpn1 and vpn2. Currently, this is impossible because both route reflectors (Device RR1 and Device RR2) learn and share information about all of the incoming VPN routes (vpn1 through vpn4) with Device PE1. In the sample topology, all devices participate in autonomous system (AS) 203, OSPF is the configured interior gateway protocol (IGP), and LDP is the signaling protocol used by the VPNs. In this example, we use static routes in the VPN routing and forwarding (VRF) instances to generate VPN routes. This is done in place of using a PE to customer edge (CE) protocol such as OSPF or BGP.

To minimize the number of VPN route updates being processed by Device PE1, you include the `family route-target proxy-generate` statement to configure proxy BGP route target filtering on each route reflector. Each route reflector has a peering session with Device PE1 and supports route target filtering to the core. However, Device PE1 does not support route target filtering, so the network resource savings are unrealized by Device PE1 since it receives all of the VPN updates. By configuring proxy BGP route target filtering on the peering sessions facing Device PE1, you limit the number of VPN updates processed by Device PE1, and the route reflectors generate the proxy BGP route target routes for Device PE1 throughout the network.

Configuration

IN THIS SECTION

CLI Quick Configuration

To quickly configure this example, copy the following commands, paste them into a text file, remove any line breaks, change any details necessary to match your network configuration, and then copy and paste the commands into the CLI at the `[edit]` hierarchy level.

Device PE1

```
set interfaces ge-1/0/0 unit 0 description PE1-to-RR1
set interfaces ge-1/0/0 unit 0 family inet address 10.49.0.1/30
set interfaces ge-1/0/0 unit 0 family mpls
set interfaces ge-1/0/1 unit 0 description PE1-to-RR2
set interfaces ge-1/0/1 unit 0 family inet address 10.49.10.1/30
set interfaces ge-1/0/1 unit 0 family mpls
set protocols ldp interface ge-1/0/0
set protocols ldp interface ge-1/0/1
set protocols bgp group internal type internal
set protocols bgp group internal local-address 10.255.163.58
set protocols bgp group internal neighbor 10.255.165.220 family inet-vpn unicast
set protocols bgp group internal neighbor 10.255.165.28 family inet-vpn unicast
set protocols ospf area 0.0.0.0 interface ge-1/0/0
set protocols ospf area 0.0.0.0 interface ge-1/0/1
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set routing-options route-distinguisher-id 10.255.163.58
set routing-options autonomous-system 203
set routing-instances vpn1 instance-type vrf
set routing-instances vpn1 vrf-target target:203:100
set routing-instances vpn1 routing-options static route 203.0.113.1/24 discard
set routing-instances vpn2 instance-type vrf
set routing-instances vpn2 vrf-target target:203:101
set routing-instances vpn2 routing-options static route 203.0.113.2/24 discard
```

Device RR1

```
set interfaces ge-1/0/0 unit 0 description RR1-to-PE1
set interfaces ge-1/0/0 unit 0 family inet address 10.49.0.2/30
```

```
set interfaces ge-1/0/0 unit 0 family mpls
set interfaces ge-1/0/1 unit 0 description RR1-to-PE2
set interfaces ge-1/0/1 unit 0 family inet address 10.50.0.2/30
set interfaces ge-1/0/1 unit 0 family mpls
set protocols ldp interface ge-1/0/0
set protocols ldp interface ge-1/0/1
set protocols bgp group internal type internal
set protocols bgp group internal local-address 198.51.100.1
set protocols bgp group internal cluster 198.51.100.1
set protocols bgp group internal neighbor 10.255.163.58 description vpn1-to-pe1 family inet-vpn
unicast
set protocols bgp group internal neighbor 10.255.163.58 family route-target proxy-generate
set protocols bgp group internal neighbor 10.255.168.42 description vpn1-to-pe2 family inet-vpn
unicast
set protocols bgp group internal neighbor 10.255.168.42 family route-target
set protocols ospf area 0.0.0.0 interface ge-1/0/0
set protocols ospf area 0.0.0.0 interface ge-1/0/1
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set routing-options route-distinguisher-id 10.255.165.220
set routing-options autonomous-system 203
```

Device RR2

```
set interfaces ge-1/0/0 unit 0 description RR2-to-PE1
set interfaces ge-1/0/0 unit 0 family inet address 10.49.10.2/30
set interfaces ge-1/0/0 unit 0 family mpls
set interfaces ge-1/0/1 unit 0 description RR2-to-PE2
set interfaces ge-1/0/1 unit 0 family inet address 10.50.10.2/30
set interfaces ge-1/0/1 unit 0 family mpls
set protocols ldp interface ge-1/0/0
set protocols ldp interface ge-1/0/1
set protocols bgp group internal type internal
set protocols bgp group internal local-address 10.255.165.28
set protocols bgp group internal cluster 198.51.100.1
set protocols bgp group internal neighbor 10.255.163.58 description vpn2-to-pe1 family inet-vpn
unicast
set protocols bgp group internal neighbor 10.255.163.58 family route-target proxy-generate
set protocols bgp group internal neighbor 10.255.168.42 description vpn2-to-pe2 family inet-vpn
unicast
set protocols bgp group internal neighbor 10.255.168.42 family route-target
set protocols ospf area 0.0.0.0 interface ge-1/0/0
set protocols ospf area 0.0.0.0 interface ge-1/0/1
```

```
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set routing-options route-distinguisher-id 10.255.165.28
set routing-options autonomous-system 203
```

Device PE2

```
set interfaces ge-1/0/0 unit 0 description PE2-to-RR1
set interfaces ge-1/0/0 unit 0 family inet address 10.50.0.1/30
set interfaces ge-1/0/0 unit 0 family mpls
set interfaces ge-1/0/1 unit 0 description PE2-to-RR2
set interfaces ge-1/0/1 unit 0 family inet address 10.50.10.1/30
set interfaces ge-1/0/1 unit 0 family mpls
set protocols ldp interface ge-1/0/0
set protocols ldp interface ge-1/0/1
set protocols bgp group internal type internal
set protocols bgp group internal local-address 10.255.168.42
set protocols bgp group internal family inet-vpn unicast
set protocols bgp group internal family route-target
set protocols bgp group internal neighbor 10.255.165.220
set protocols bgp group internal neighbor 10.255.165.28
set protocols ospf area 0.0.0.0 interface ge-1/0/0
set protocols ospf area 0.0.0.0 interface ge-1/0/1
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set routing-options route-distinguisher-id 10.255.168.42
set routing-options autonomous-system 203
set routing-instances vpn1 instance-type vrf
set routing-instances vpn1 vrf-target target:203:100
set routing-instances vpn1 routing-options static route 203.0.113.1/24 discard
set routing-instances vpn2 instance-type vrf
set routing-instances vpn2 vrf-target target:203:101
set routing-instances vpn2 routing-options static route 203.0.113.2/24 discard
set routing-instances vpn3 instance-type vrf
set routing-instances vpn3 vrf-target target:203:103
set routing-instances vpn3 routing-options static route 203.0.113.3/24 discard
set routing-instances vpn4 instance-type vrf
set routing-instances vpn4 vrf-target target:203:104
set routing-instances vpn4 routing-options static route 203.0.113.4/24 discard
```

Configuring Device PE1

Step-by-Step Procedure

The following example requires you to navigate various levels in the configuration hierarchy. For information about navigating the CLI, see *Using the CLI Editor in Configuration Mode*.

To configure Device PE1:

1. Configure the interfaces.

   ```
   [edit interfaces]
   user@PE1# set ge-1/0/0 unit 0 description PE1-to-RR1
   user@PE1# set ge-1/0/0 unit 0 family inet address 10.49.0.1/30
   user@PE1# set ge-1/0/0 unit 0 family mpls
   user@PE1# set ge-1/0/1 unit 0 description PE1-to-RR2
   user@PE1# set ge-1/0/1 unit 0 family inet address 10.49.10.1/30
   user@PE1# set ge-1/0/1 unit 0 family mpls
   ```

2. Configure the route distinguisher and the AS number.

   ```
   [edit routing-options]
   user@PE1# set route-distinguisher-id 10.255.163.58
   user@PE1# set autonomous-system 203
   ```

3. Configure LDP as the signaling protocol used by the VPN.

   ```
   [edit protocols ldp]
   user@PE1# set interface ge-1/0/0
   user@PE1# set interface ge-1/0/1
   ```

4. Configure BGP.

   ```
   [edit protocols bgp group internal]
   user@PE1# set type internal
   user@PE1# set local-address 10.255.163.58
   user@PE1# set neighbor 10.255.165.220 family inet-vpn unicast
   user@PE1# set neighbor 10.255.165.28 family inet-vpn unicast
   ```

5. Configure OSPF.

```
[edit protocols ospf area 0.0.0.0]
user@PE1# set interface ge-1/0/0
user@PE1# set interface ge-1/0/1
user@PE1# set interface lo0.0 passive
```

6. Configure the VPN routing instances.

```
[edit routing-instances vpn1]
user@PE1# set instance-type vrf
user@PE1# set vrf-target target:203:100
user@PE1# set routing-options static route 203.0.113.1/24 discard
```

```
[edit routing-instances vpn2]
user@PE1# set instance-type vrf
user@PE1# set vrf-target target:203:101
user@PE1# set routing-options static route 203.0.113.2/24 discard
```

7. If you are done configuring the device, commit the configuration.

```
[edit]
user@PE1# commit
```

Results

From configuration mode, confirm your configuration by entering the `show interfaces`, `show protocols`, `show routing-options`, and `show routing-instances` commands. If the output does not display the intended configuration, repeat the instructions in this example to correct the configuration.

```
user@PE1# show interfaces
ge-1/0/0 {
    unit 0 {
        description PE1-to-RR1;
        family inet {
            address 10.49.0.1/30;
        }
```

```
        family mpls;
    }
}
ge-1/0/1 {
    unit 0 {
        description PE1-to-RR2;
        family inet {
            address 10.49.10.1/30;
        }
        family mpls;
    }
}
```

```
user@PE1# show protocols
bgp {
    group internal {
        type internal;
        local-address 10.255.163.58;
        neighbor 10.255.165.220 {
            family inet-vpn {
                unicast;
            }
        }
        neighbor 10.255.165.28 {
            family inet-vpn {
                unicast;
            }
        }
    }
}
ospf {
    area 0.0.0.0 {
        interface ge-1/0/0.0;
        interface ge-1/0/1.0;
        interface lo0.0 {
            passive;
        }
    }
}
ldp {
    interface ge-1/0/0.0;
```

```
    interface ge-1/0/1.0;
}
```

```
user@PE1# show routing-options
route-distinguisher-id 10.255.14.182;
autonomous-system 203;
```

```
user@PE1# show routing-instances
vpn1 {
    instance-type vrf;
    vrf-target target:203:100;
    routing-options {
        static {
            route 203.0.113.1/24 discard;
        }
    }
}
vpn2 {
    instance-type vrf;
    vrf-target target:203:101;
    routing-options {
        static {
            route 203.0.113.2/24 discard;
        }
    }
}
```

Configuring Device RR1

Step-by-Step Procedure

The following example requires you to navigate various levels in the configuration hierarchy. For information about navigating the CLI, see *Using the CLI Editor in Configuration Mode*.

To configure Device RR1:

1. Configure the interfaces.

```
[edit interfaces]
user@RR1# set ge-1/0/0 unit 0 description RR1-to-PE1
user@RR1# set ge-1/0/0 unit 0 family inet address 10.49.0.2/30
user@RR1# set ge-1/0/0 unit 0 family mpls
user@RR1# set ge-1/0/1 unit 0 description RR1-to-PE2
user@RR1# set ge-1/0/1 unit 0 family inet address 10.50.0.2/30
user@RR1# set ge-1/0/1 unit 0 family mpls
```

2. Configure the route distinguisher and the AS number.

```
[edit routing-options]
user@RR1# set route-distinguisher-id 10.255.165.220
user@RR1# set autonomous-system 203
```

3. Configure LDP as the signaling protocol used by the VPN.

```
[edit protocols ldp]
user@RR1# set interface ge-1/0/0
user@RR1# set interface ge-1/0/1
```

4. Configure BGP.

```
[edit protocols bgp group internal]
user@RR1# set type internal
user@RR1# set local-address 10.255.165.220
user@RR1# set cluster 198.51.100.1
user@RR1# set neighbor 10.255.163.58 description vpn1-to-pe1 family inet-vpn unicast
user@RR1# set neighbor 10.255.168.42 description vpn1-to-pe2 family inet-vpn unicast
```

5. Configure BGP route target filtering on the peering session with Device PE2.

```
[edit protocols bgp group internal]
user@RR1# set neighbor 10.255.168.42 family route-target
```

6. Configure proxy BGP route target filtering on the peering session with Device PE1.

```
[edit protocols bgp group internal]
user@RR1# set neighbor 10.255.163.58 family route-target proxy-generate
```

7. Configure OSPF.

```
[edit protocols ospf area 0.0.0.0]
user@RR1# set interface ge-1/0/0
user@RR1# set interface ge-1/0/1
user@RR1# set interface lo0.0 passive
```

8. If you are done configuring the device, commit the configuration.

```
[edit]
user@RR1# commit
```

Results

From configuration mode, confirm your configuration by entering the `show interfaces`, `show protocols` and `show routing-options` commands. If the output does not display the intended configuration, repeat the instructions in this example to correct the configuration.

```
user@RR1# show interfaces
ge-1/0/0 {
    unit 0 {
        description RR1-to-PE1;
        family inet {
            address 10.49.0.2/30;
        }
        family mpls;
    }
}
ge-1/0/1 {
    unit 0 {
        description RR1-to-PE2;
        family inet {
            address 10.50.0.2/30;
```

```
        }
        family mpls;
    }
}
```

```
user@RR1# show protocols
bgp {
    group internal {
        type internal;
        local-address 198.51.100.1;
        cluster 198.51.100.1;
        neighbor 10.255.163.58 {
            description vpn1-to-pe1;
            family inet-vpn {
                unicast;
            }
            family route-target {
                proxy-generate;
            }
        }
        neighbor 10.255.168.42 {
            description vpn1-to-pe2;
            family inet-vpn {
                unicast;
            }
            family route-target;
        }
    }
}
ospf {
    area 0.0.0.0 {
        interface ge-1/0/0.0;
        interface ge-1/0/1.0;
        interface lo0.0 {
            passive;
        }
    }
}
ldp {
    interface ge-1/0/0.0;
```

```
    interface ge-1/0/1.0;
}
```

```
user@RR1# show routing-options
route-distinguisher-id 10.255.165.220;
autonomous-system 203;
```

Configuring Device RR2

Step-by-Step Procedure

The following example requires you to navigate various levels in the configuration hierarchy. For information about navigating the CLI, see *Using the CLI Editor in Configuration Mode*.

To configure Device RR2:

1. Configure the interfaces.

   ```
   [edit interfaces]
   user@RR2# set ge-1/0/0 unit 0 description RR2-to-PE1
   user@RR2# set ge-1/0/0 unit 0 family inet address 10.49.10.2/30
   user@RR2# set ge-1/0/0 unit 0 family mpls
   user@RR2# set ge-1/0/1 unit 0 description RR2-to-PE2
   user@RR2# set ge-1/0/1 unit 0 family inet address 10.50.10.2/30
   user@RR2# set ge-1/0/1 unit 0 family mpls
   ```

2. Configure the route distinguisher and the AS number.

   ```
   [edit routing-options]
   user@RR2# set route-distinguisher-id 10.255.165.28
   user@RR2# set autonomous-system 203
   ```

3. Configure LDP as the signaling protocol used by the VPN.

   ```
   [edit protocols ldp]
   user@RR2# set interface ge-1/0/0
   user@RR2# set interface ge-1/0/1
   ```

4. Configure BGP.

```
[edit protocols bgp group internal]
user@RR2# set type internal
user@RR2# set local-address 10.255.165.28
user@RR2# set cluster 198.51.100.1
user@RR2# set neighbor 10.255.163.58 description vpn2-to-pe1 family inet-vpn unicast
user@RR2# set neighbor 10.255.168.42 description vpn2-to-pe2 family inet-vpn unicast
```

5. Configure BGP route target filtering on the peering session with Device PE2.

```
[edit protocols bgp group internal]
user@RR2# set neighbor 10.255.168.42 family route-target
```

6. Configure proxy BGP route target filtering on the peering session with Device PE1.

```
[edit protocols bgp group internal]
user@RR2# set neighbor 10.255.163.58 family route-target proxy-generate
```

7. Configure OSPF.

```
[edit protocols ospf area 0.0.0.0]
user@RR2# set interface ge-1/0/0
user@RR2# set interface ge-1/0/1
user@RR2# set interface lo0.0 passive
```

8. If you are done configuring the device, commit the configuration.

```
[edit]
user@RR2# commit
```

Results

From configuration mode, confirm your configuration by entering the `show interfaces`, `show protocols`, and `show routing-options` commands. If the output does not display the intended configuration, repeat the instructions in this example to correct the configuration.

```
user@RR2# show interfaces
ge-1/0/0 {
    unit 0 {
        description RR2-to-PE1;
        family inet {
            address 10.49.10.2/30;
        }
        family mpls;
    }
}
ge-1/0/1 {
    unit 0 {
        description RR2-to-PE2;
        family inet {
            address 10.50.10.2/30;
        }
        family mpls;
    }
}
```

```
user@RR2# show protocols
bgp {
    group internal {
        local-address 10.255.165.28;
        cluster 198.51.100.1;
        neighbor 10.255.163.58 {
            description vpn2-to-pe1;
            family inet-vpn {
                unicast;
            }
            family route-target {
                proxy-generate;
            }
        }
        neighbor 10.255.168.42 {
```

```
            description vpn2-to-pe2;
            family inet-vpn {
                unicast;
            }
            family route-target;
        }
    }
}
ospf {
    area 0.0.0.0 {
        interface ge-1/0/0.0;
        interface ge-1/0/1.0;
        interface lo0.0 {
            passive;
        }
    }
}
ldp {
    interface ge-1/0/0.0;
    interface ge-1/0/1.0;
}
```

```
user@RR2# show routing-options
route-distinguisher-id 10.255.165.28;
autonomous-system 203;
```

Configuring Device PE2

Step-by-Step Procedure

The following example requires you to navigate various levels in the configuration hierarchy. For information about navigating the CLI, see *Using the CLI Editor in Configuration Mode*.

To configure Device PE2:

1. Configure the interfaces.

```
[edit interfaces]
user@PE2# set ge-1/0/0 unit 0 description PE2-to-RR1
user@PE2# set ge-1/0/0 unit 0 family inet address 10.50.0.1/30
user@PE2# set ge-1/0/0 unit 0 family mpls
```

```
user@PE2# set ge-1/0/1 unit 0 description PE2-to-RR2
user@PE2# set ge-1/0/1 unit 0 family inet address 10.50.10.1/30
user@PE2# set ge-1/0/1 unit 0 family mpls
```

2. Configure the route distinguisher and the AS number.

```
[edit routing-options]
user@PE2# set route-distinguisher-id 10.255.168.42
user@PE2# set autonomous-system 203
```

3. Configure LDP as the signaling protocol used by the VPN.

```
[edit protocols ldp]
user@PE2# set interface ge-1/0/0
user@PE2# set interface ge-1/0/1
```

4. Configure BGP.

```
[edit protocols bgp group internal]
user@PE2# set type internal
user@PE2# set local-address 10.255.168.42
user@PE2# set family inet-vpn unicast
user@PE2# set family route-target
user@PE2# set neighbor 10.255.165.220
user@PE2# set neighbor 10.255.165.28
```

5. Configure OSPF.

```
[edit protocols ospf area 0.0.0.0]
user@PE2# set interface ge-1/0/0
user@PE2# set interface ge-1/0/1
user@PE2# set interface lo0.0 passive
```

6. Configure the VPN routing instances.

```
[edit routing-instances vpn1]
user@PE2# set instance-type vrf
```

```
user@PE2# set vrf-target target:203:100
user@PE2# set routing-options static route 203.0.113.1/24 discard
```

```
[edit routing-instances vpn2]
user@PE2# set instance-type vrf
user@PE2# set vrf-target target:203:101
user@PE2# set routing-options static route 203.0.113.2/24 discard
```

```
[edit routing-instances vpn3]
user@PE2# set instance-type vrf
user@PE2# set vrf-target target:203:103
user@PE2# set routing-options static route 203.0.113.3/24 discard
```

```
[edit routing-instances vpn4]
user@PE2# set instance-type vrf
user@PE2# set vrf-target target:203:104
user@PE2# set routing-options static route 203.0.113.4/24 discard
```

7. If you are done configuring the device, commit the configuration.

```
[edit]
user@PE2# commit
```

Results

From configuration mode, confirm your configuration by entering the `show interfaces`, `show protocols`, `show routing-options`, and `show routing-instances` commands. If the output does not display the intended configuration, repeat the instructions in this example to correct the configuration.

```
user@PE2# show interfaces
ge-1/0/0 {
    unit 0 {
        description PE2-to-RR1;
        family inet {
            address 10.50.0.1/30;
        }
```

```
        family mpls;
    }
}
ge-1/0/1 {
    unit 0 {
        description PE2-to-RR2;
        family inet {
            address 10.50.10.1/30;
        }
        family mpls;
    }
}
```

```
user@PE2# show protocols
bgp {
    group internal {
        type internal;
        local-address 10.255.168.42;
        family inet-vpn {
            unicast;
        }
        family route-target;
        neighbor 10.255.165.220;
        neighbor 10.255.165.28;
    }
}
ospf {
    area 0.0.0.0 {
        interface ge-1/0/0.0;
        interface ge-1/0/1.0;
        interface lo0.0 {
            passive;
        }
    }
}
ldp {
    interface ge-1/0/0.0;
```

```
    interface ge-1/0/1.0;
}
```

```
user@PE2# show routing-options
route-distinguisher-id 10.255.168.42;
autonomous-system 203;
```

```
user@PE2# show routing-instances
vpn1 {
    instance-type vrf;
    vrf-target target:203:100;
    routing-options {
        static {
            route 203.0.113.1/24 discard;
        }
    }
}
vpn2 {
    instance-type vrf;
    vrf-target target:203:101;
    routing-options {
        static {
            route 203.0.113.2/24 discard;
        }
    }
}
vpn3 {
    instance-type vrf;
    vrf-target target:203:103;
    routing-options {
        static {
            route 203.0.113.3/24 discard;
        }
    }
}
vpn4 {
    instance-type vrf;
    vrf-target target:203:104;
    routing-options {
        static {
```

```
            route 203.0.113.4/24 discard;
        }
    }
}
```

Verification

IN THIS SECTION

Confirm that the configuration is working properly.

Verifying the Proxy BGP Route Target Routes

Purpose

Verify that the proxy BGP route target routes are displayed in the bgp.rtarget.0 table on Device RR1.

Action

From operational mode, enter the `show route table bgp.rtartget.0` command to display the proxy BGP route targets.

```
user@RR1# show route table bgp.rtarget.0
4 destinations, 6 routes (4 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

203:203:100/96
                   *[RTarget/5] 00:01:22
                      Type Proxy
                        for 10.255.163.58
                         Local
                    [BGP/170] 00:04:55, localpref 100, from 10.255.168.42
                      AS path: I, validation-state: unverified
                    > to 10.50.0.1 via ge-1/0/1
203:203:101/96
```

```
                   *[RTarget/5] 00:01:22
                      Type Proxy
                        for 10.255.163.58
                         Local
                    [BGP/170] 00:04:55, localpref 100, from 10.255.168.42
                      AS path: I, validation-state: unverified
                    > to 10.50.0.1 via ge-1/0/1
203:203:103/96
                   *[BGP/170] 00:04:55, localpref 100, from 10.255.168.42
                      AS path: I, validation-state: unverified
                    > to 10.50.0.1 via ge-1/0/1
203:203:104/96
                   *[BGP/170] 00:04:55, localpref 100, from 10.255.168.42
                      AS path: I, validation-state: unverified
                    > to 10.50.0.1 via ge-1/0/1
```

Meaning

Device RR1 is generating the proxy BGP route target routes on behalf of its peer Device PE1. The proxy BGP route target routes are identified with the protocol and preference `[RTarget/5]` and the route target type of `Proxy`.

RELATED DOCUMENTATION

Understanding Route Filters for Use in Routing Policy Match Conditions

Route Filter Match Conditions

Example: Configuring Policy Chains and Route Filters

Example: Configuring the MED Using Route Filters

Example: Configuring a Route Filter Policy to Specify Priority for Prefixes Learned Through OSPF

Configuring Routing Between PE and CE Routers

IN THIS SECTION

This topic provides information on how to configure routing on PE and CE routers in a Layer 3 VPN.

Configuring Routing Between PE and CE Routers in Layer 3 VPNs

IN THIS SECTION

For the PE router to distribute VPN-related routes to and from connected CE routers, you must configure routing within the VPN routing instance. You can configure a routing protocol—BGP, OSPF, or RIP—or you can configure static routing. For the connection to each CE router, you typically configure one type of routing, but in some cases you can include both static routes and routing protocol configurations.

The following sections explain how to configure VPN routing between the PE and CE routers:

Configuring BGP Between the PE and CE Routers

To configure BGP as the routing protocol between the PE and the CE routers, include the `bgp` statement:

```
bgp {
    group group-name {
        peer-as as-number;
        neighbor ip-address;
    }
}
```

You can include this statement at the following hierarchy levels:

- `[edit routing-instances` *`routing-instance-name`* `protocols]`
- `[edit logical-systems` *`logical-system-name`* `routing-instances` *`routing-instance-name`* `protocols]`

> **NOTE**: The `[edit logical-systems]` hierarchy level is not applicable in ACX Series routers.

Please be aware of the following limitations regarding configuring BGP for routing instances:

- In a VRF routing instance, do not configure the local autonomous system (AS) number using an AS number that is already in use by a remote BGP peer in a separate VRF routing instance. Doing so creates an autonomous system loop where all the routes received from this remote BGP peer are hidden.

  You configure the local AS number using either the `autonomous-system` statement at the `[edit routing-instances` *`routing-instance-name`* `routing-options]` hierarchy level or the `local-as` statement at any of the following hierarchy levels:

  - `[edit routing-instances` *`routing-instance-name`* `protocols bgp]`
  - `[edit routing-instances` *`routing-instance-name`* `protocols bgp group` *`group-name`*`]`
  - `[edit routing-instances` *`routing-instance-name`* `protocols bgp group` *`group-name`* `neighbor` *`address`*`]`

  You configure the AS number for a BGP peer using the `peer-as` statement at the `[edit routing-instances` *`routing-instance-name`* `protocols bgp group` *`group-name`*`]` hierarchy level.

Configuring OSPF Between the PE and CE Routers

You can configure OSPF (version 2 or version 3) to distribute VPN-related routes between PE and CE routers.

The following sections describe how to configure OSPF as a routing protocol between the PE and the CE routers:

Configuring OSPF Version 2 Between the PE and CE Routers

To configure OSPF version 2 as the routing protocol between a PE and CE router, include the `ospf` statement:

```
ospf {
    area area {
        interface interface-name;
    }
}
```

You can include this statement at the following hierarchy levels:

- `[edit routing-instances` *`routing-instance-name`* `protocols]`
- `[edit logical-systems` *`logical-system-name`* `routing-instances` *`routing-instance-name`* `protocols]`

NOTE: The `[edit logical-systems]` hierarchy level is not applicable in ACX Series routers.

Configuring OSPF Version 3 Between the PE and CE Routers

To configure OSPF version 3 as the routing protocol between a PE and CE router, include the `ospf3` statement:

```
ospf3 {
    area area {
        interface interface-name;
    }
}
```

You can include this statement at the following hierarchy levels:

- `[edit routing-instances` *`routing-instance-name`* `protocols]`
- `[edit logical-systems` *`logical-system-name`* `routing-instances` *`routing-instance-name`* `protocols]`

NOTE: The `[edit logical-systems]` hierarchy level is not applicable in ACX Series routers.

Configuring OSPF Sham Links for Layer 3 VPNs

When you configure OSPF between the PE and CE routers of a Layer 3 VPN, you can also configure OSPF sham links to compensate for issues related to OSPF intra-area links.

The following sections describe OSPF sham links and how to configure them:

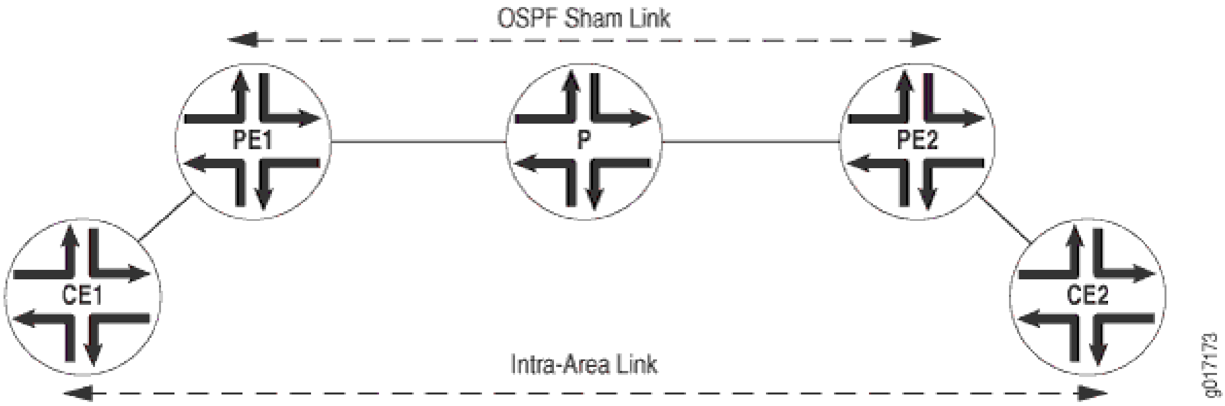
OSPF Sham Links Overview

Figure 13 on page 160 provides an illustration of when you might configure an OSPF sham link. Router CE1 and Router CE2 are located in the same OSPF area. These CE routers are linked together by a Layer 3 VPN over Router PE1 and Router PE2. In addition, Router CE1 and Router CE2 are connected by an intra-area link used as a backup.

OSPF treats the link through the Layer 3 VPN as an interarea link. By default, OSPF prefers intra-area links to interarea links, so OSPF selects the backup intra-area link as the active path. This is not acceptable in configurations where the intra-area link is not the expected primary path for traffic between the CE routers.

An OSPF sham link is also an intra-area link, except that it is configured between the PE routers as shown in Figure 13 on page 160. You can configure the metric for the sham link to ensure that the path over the Layer 3 VPN is preferred to a backup path over an intra-area link connecting the CE routers.

Figure 13: OSPF Sham Link



You should configure an OSPF sham link under the following circumstances:

- Two CE routers are linked together by a Layer 3 VPN.

- These CE routers are in the same OSPF area.
- An intra-area link is configured between the two CE routers.

If there is no intra-area link between the CE routers, you do not need to configure an OSPF sham link.

For more information about OSPF sham links, see the Internet draft draft-ietf-l3vpn-ospf-2547-01.txt, *OSPF as the PE/CE Protocol in BGP/MPLS VPNs*.

Configuring OSPF Sham Links

The sham link is an unnumbered point-to-point intra-area link and is advertised by means of a type 1 link-state advertisement (LSA). Sham links are valid only for routing instances and OSPF version 2.

Each sham link is identified by a combination of the local and remote sham link end-point address and the OSPF area to which it belongs. Sham links must be configured manually. You configure the sham link between two PE routers, both of which are within the same VRF routing instance.

You need to specify the address for the local end point of the sham link. This address is used as the source for the sham link packets and is also used by the remote PE router as the sham link remote end-point.

The OSPF sham link's local address must be specified with a loopback address for the local VPN. The route to this address must be propagated by BGP. Specify the address for the local end point using the **local** option of the **sham-link** statement:

```
sham-link {
    local address;
}
```

You can include this statement at the following hierarchy levels:

- **[edit routing-instances *routing-instance-name* protocols ospf]**
- **[edit logical-systems *logical-system-name* routing-instances *routing-instance-name* protocols ospf]**

The OSPF sham link's remote address must be specified with a loopback address for the remote VPN. The route to this address must be propagated by BGP. To specify the address for the remote end point, include the **sham-link-remote** statement:

```
sham-link-remote address <metric number>;
```

You can include this statement at the following hierarchy levels:

- **[edit routing-instances *routing-instance-name* protocols ospf area *area-id*]**

- **[edit logical-systems *logical-system-name* routing-instances *routing-instance-name* protocols ospf area *area-id*]**

Optionally, you can include the **metric** option to set a metric value for the remote end point. The metric value specifies the cost of using the link. Routes with lower total path metrics are preferred over those with higher path metrics.

You can configure a value from 1 through 65,535. The default value is 1.

OSPF Sham Links Example

This example shows how to enable OSPF sham links on a PE router.

The following is the loopback interface configuration on the PE router. The address configured is for the local end point of the OSPF sham link:

```
[edit]
interfaces {
    lo0 {
        unit 1 {
            family inet {
                address 10.1.1.1/32;
            }
        }
    }
}
```

The following is the routing instance configuration on the PE router, including the configuration for the OSPF sham link. The **sham-link local** statement is configured with the address for the local loopback interface:

```
[edit]
routing-instances {
    example-sham-links {
        instance-type vrf;
        interface e1-1/0/2.0;
        interface lo0.1;
        route-distinguisher 3:4;
        vrf-import vpn-red-import;
        vrf-export vpn-red-export;
        protocols {
            ospf {
```

```
                sham-link local 10.1.1.1;
                area 0.0.0.0 {
                    sham-link-remote 10.2.2.2 metric 1;
                    interface e1-1/0/2.0 metric 1;
                }
            }
        }
    }
}
```

Configuring an OSPF Domain ID

For most OSPF configurations involving Layer 3 VPNs, you do not need to configure an OSPF domain ID. However, for a Layer 3 VPN connecting multiple OSPF domains, configuring OSPF domain IDs can help you control LSA translation (for Type 3 and Type 5 LSAs) between the OSPF domains and back-door paths. Each VPN routing and forwarding (VRF) table in a PE router associated with an OSPF instance is configured with the same OSPF domain ID. The default OSPF domain ID is the null value 0.0.0.0. As shown in Table 6 on page 163, a route with a null domain ID is handled differently from a route without any domain ID at all.

Table 6: How a PE Router Redistributes and Advertises Routes

Route Received	Domain ID of the Route Received	Domain ID on the Receiving Router	Route Redistributed and Advertised As
Type 3 route	A.B.C.D	A.B.C.D	Type 3 LSA
Type 3 route	A.B.C.D	E.F.G.H	Type 5 LSA
Type 3 route	0.0.0.0	0.0.0.0	Type 3 LSA
Type 3 route	Null	0.0.0.0	Type 3 LSA
Type 3 route	Null	Null	Type 3 LSA
Type 3 route	0.0.0.0	Null	Type 3 LSA
Type 3 route	A.B.C.D	Null	Type 5 LSA

Table 6: How a PE Router Redistributes and Advertises Routes *(Continued)*

Route Received	Domain ID of the Route Received	Domain ID on the Receiving Router	Route Redistributed and Advertised As
Type 3 route	Null	A.B.C.D	Type 3 LSA
Type 5 route	Not applicable	Not applicable	Type 5 LSA

You can configure an OSPF domain ID for both version 2 and version 3 of OSPF. The only difference in the configuration is that you include statements at the **[edit routing-instances *routing-instance-name* protocols ospf]** hierarchy level for OSPF version 2 and at the **[edit routing-instances *routing-instance-name* protocols ospf3]** hierarchy level for OSPF version 3. The configuration descriptions that follow present the OSPF version 2 statement only. However, the substatements are also valid for OSPF version 3.

To configure an OSPF domain ID, include the **domain-id** statement:

```
domain-id domain-Id;
```

You can include this statement at the following hierarchy levels:

- **[edit routing-instances *routing-instance-name* protocols ospf]**
- **[edit logical-systems *logical-system-name* routing-instances *routing-instance-name* protocols ospf]**

You can set a VPN tag for the OSPF external routes generated by the PE router to prevent looping. By default, this tag is automatically calculated and needs no configuration. However, you can configure the domain VPN tag for Type 5 LSAs explicitly by including the **domain-vpn-tag** statement:

```
no-domain-vpn-tag number;
```

You can include this statement at the following hierarchy levels:

- **[edit routing-instances *routing-instance-name* protocols ospf]**
- **[edit logical-systems *logical-system-name* routing-instances *routing-instance-name* protocols ospf]**

The range is 1 through 4,294,967,295 (2^{32} – 1). If you set VPN tags manually, you must set the same value for all PE routers in the VPN.

For an example of this type of configuration, see "Configuring an OSPF Domain ID for a Layer 3 VPN" on page 169.

Hub-and-Spoke Layer 3 VPNs and OSPF Domain IDs

The default behavior of an OSPF domain ID causes some problems for hub-and-spoke Layer 3 VPNs configured with OSPF between the hub PE router and the hub CE router when the routes are not aggregated. A hub-and-spoke configuration has a hub PE router with direct links to a hub CE router. The hub PE router receives Layer 3 BGP updates from the other remote spoke PE routers, and these are imported into the spoke routing instance. From the spoke routing instance, the OSPF LSAs are originated and sent to the hub CE router.

The hub CE router typically aggregates these routes, and then sends these newly originated LSAs back to the hub PE router. The hub PE router exports the BGP updates to the remote spoke PE routers containing the aggregated prefixes. However, if there are nonaggregated Type 3 summary LSAs or external LSAs, two issues arise with regard to how the hub PE router originates and sends LSAs to the hub CE router, and how the hub PE router processes LSAs received from the hub CE router:

- By default, all LSAs originated by the hub PE router in the spoke routing instance have the DN bit set. Also, all externally originated LSAs have the VPN route tag set. Setting the DN bit and the VPN route tag help prevent routing loops. For Type 3 summary LSAs, routing loops are not a concern because the hub CE router, as an area border router (ABR), reoriginates the LSAs with the DN bit clear and sends them back to the hub PE router. However, the hub CE router does not reoriginate external LSAs, because they have an AS flooding scope.

  You can originate the external LSAs (before sending them to the hub CE router) with the DN bit clear and the VPN route tag set to 0 by altering the hub PE router's routing instance configuration. To clear the DN bit and set the VPN route tag to zero on external LSAs originated by a PE router, configure 0 for the **domain-vpn-tag** statement at the **[edit routing-instances *routing-instance-name* protocols ospf]** hierarchy level. You should include this configuration in the routing instance on the hub PE router facing the hub CE router where the LSAs are sent. When the hub CE router receives external LSAs from the hub PE router and then forwards them back to the hub PE router, the hub PE router can use the LSAs in its OSPF route calculation.

- When LSAs flooded by the hub CE router arrive at the hub PE router's routing instance, the hub PE router, acting as an ABR, does not consider these LSAs in its OSPF route calculations, even though the LSAs do not have the DN bits set and the external LSAs do not have a VPN route tag set. The LSAs are assumed to be from a disjoint backbone area.

  You can change the configuration of the PE router's routing instance to cause the PE router to act as a non-ABR by including the **disable** statement at the **[edit routing-instances *routing-instance-name* protocols ospf domain-id]** hierarchy level. You make this configuration change to the hub PE router that receives the LSAs from the hub CE router.

  By making this configuration change, the PE router's routing instance acts as a non-ABR. The PE router then considers the LSAs arriving from the hub CE router as if they were coming from a contiguous nonbackbone area.

Configuring RIP Between the PE and CE Routers

For a Layer 3 VPN, you can configure RIP on the PE router to learn the routes of the CE router or to propagate the routes of the PE router to the CE router. RIP routes learned from neighbors configured at any `[edit routing-instances]` hierarchy level are added to the routing instance's `inet` table (`*instance_name*.inet.0`).

To configure RIP as the routing protocol between the PE and the CE router, include the `rip` statement:

```
rip {
    group group-name {
        export policy-names;
        neighbor interface-name;
    }
}
```

You can include this statement at the following hierarchy levels:

- `[edit routing-instances routing-instance-name protocols]`
- `[edit logical-systems logical-system-name routing-instances routing-instance-name protocols]`

> **NOTE**: The `[edit logical-systems]` hierarchy level is not applicable in ACX Series routers.

By default, RIP does not advertise the routes it receives. To advertise routes from a PE router to a CE router, you need to configure an export policy on the PE router for RIP. For information about how to define policies for RIP, see *RIP Import Policy*.

To specify an export policy for RIP, include the `export` statement:

```
export [ policy-names ];
```

You can include this statement for RIP at the following hierarchy levels:

- `[edit routing-instances routing-instance-name protocols rip group group-name]`
- `[edit logical-systems logical-system-name routing-instances routing-instance-name protocols rip group group-name]`

> **NOTE**: The `[edit logical-systems]` hierarchy level is not applicable in ACX Series routers.

To install routes learned from a RIP routing instance into multiple routing tables, include the `rib-group` and `group` statements:

```
rib-group inet group-name;
group group-name {
    neighbor interface-name;
}
```

You can include these statements at the following hierarchy levels:

- `[edit protocols rip]`
- `[edit routing-instances routing-instance-name protocols rip]`
- `[edit logical-systems logical-system-name protocols rip]`
- `[edit logical-systems logical-system-name routing-instances routing-instance-name protocols rip]`

NOTE: The `[edit logical-systems]` hierarchy level is not applicable in ACX Series routers.

To configure a routing table group, include the `rib-groups` statement:

```
rib-groups group-name;
```

You can include this statement at the following hierarchy levels:

- `[edit routing-options]`
- `[edit logical-systems logical-system-name routing-options]`

NOTE: The `[edit logical-systems]` hierarchy level is not applicable in ACX Series routers.

To add a routing table to a routing table group, include the `import-rib` statement. The first routing table name specified under the `import-rib` statement must be the name of the routing table you are configuring. For more information about how to configure routing tables and routing table groups, see Junos OS Routing Protocols Library.

```
import-rib [ group-names ];
```

You can include this statement at the following hierarchy levels:

- `[edit routing-options rib-groups` *`group-name`*`]`
- `[edit logical-systems` *`logical-system-name`* `routing-options rib-groups` *`group-name`*`]`

NOTE: The `[edit logical-systems]` hierarchy level is not applicable in ACX Series routers.

RIP instances are supported only for VRF instance types. You can configure multiple instances of RIP for VPN support only. You can use RIP in the customer edge-provider edge (CE-PE) environment to learn routes from the CE router and to propagate the PE router's instance routes in the CE router.

RIP routes learned from neighbors configured under any instance hierarchy are added to the instance's routing table, *`instance-name`*`.inet.0`.

RIP does not support routing table groups; therefore, it cannot import routes into multiple tables as the OSPF or OSPFv3 protocol does.

Configuring Static Routes Between the PE and CE Routers

You can configure static (nonchanging) routes between the PE and CE routers of a VPN routing instance. To configure a static route for a VPN, you need to configure it within the VPN routing instance configuration at the `[edit routing-instances` *`routing-instance-name`* `routing-options]` hierarchy level.

To configure a static route between the PE and the CE routers, include the `static` statement:

```
static {
    route destination-prefix {
        next-hop [ next-hops ];
        static-options;
    }
}
```

You can include this statement at the following hierarchy levels:

- `[edit routing-instances` *`routing-instance-name`* `routing-options]`
- `[edit logical-systems` *`logical-system-name`* `routing-instances` *`routing-instance-name`* `routing-options]`

NOTE: The `[edit logical-systems]` hierarchy level is not applicable in ACX Series routers.

For more information about configuring routing protocols and static routes, see Junos OS Routing Protocols Library.

Configuring an OSPF Domain ID for a Layer 3 VPN

IN THIS SECTION

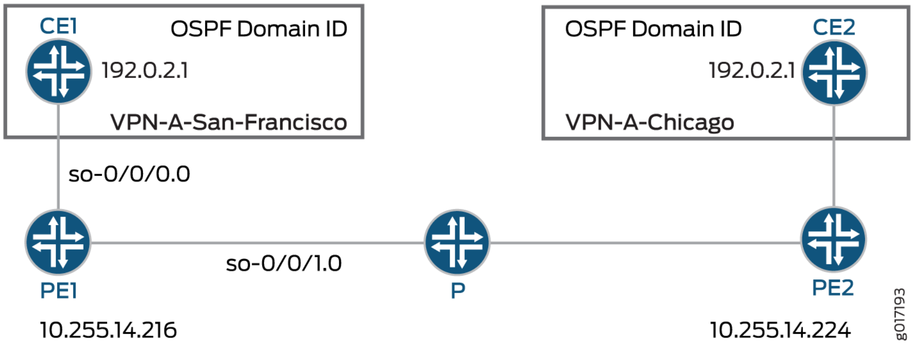
This example illustrates how to configure an OSPF domain ID for a VPN by using OSPF as the routing protocol between the PE and CE routers. Routes from an OSPF domain need an OSPF domain ID when they are distributed in BGP as VPN-IPv4 routes in VPNs with multiple OSPF domains. In a VPN connecting multiple OSPF domains, the routes from one domain might overlap with the routes of another.

The domain ID that is configured in a routing instance identifies the OSPF domain and is used to identify the route origination. The domain ID that is configured on a community policy is used in setting exported routes.

For more information about OSPF domain IDs and Layer 3 VPNs, see "Configuring Routing Between PE and CE Routers in Layer 3 VPNs" on page 157.

Figure 14 on page 170 shows this example's configuration topology. Only the configuration for Router PE1 is provided. The configuration for Router PE2 can be similar to the configuration for Router PE1. There are no special configuration requirements for the CE routers.

Figure 14: Example of a Configuration Using an OSPF Domain ID



For configuration information, see the following sections:

Configuring Interfaces on Router PE1

You need to configure two interfaces for Router PE1—the `so-0/0/0` interface for traffic to Router CE1 (San Francisco) and the `so-0/0/1` interface for traffic to a P router in the service provider's network.

Configure the interfaces for Router PE1:

```
[edit]
interfaces {
    so-0/0/0 {
        unit 0 {
            family inet {
                address 10.19.1.2/30;
            }
        }
    }
    so-0/0/1 {
        unit 0 {
            family inet {
                address 10.19.2.1/30;
            }
            family mpls;
        }
    }
}
```

Configuring Routing Options on Router PE1

At the `[edit routing-options]` hierarchy level, you need to configure the `router-id` and `autonomous-system` statements. The `router-id` statement identifies Router PE1.

Configure the routing options for Router PE1:

```
[edit]
routing-options {
    router-id 10.255.14.216;
    autonomous-system 65069;
}
```

Configuring Protocols on Router PE1

On Router PE1, you need to configure MPLS, BGP, OSPF, and LDP at the `[edit protocols]` hierarchy level:

```
[edit]
protocols {
    mpls {
        interface so-0/0/1.0;
    }
    bgp {
        group San-Francisco-Chicago {
            type internal;
            preference 10;
            local-address 10.255.14.216;
            family inet-vpn {
                unicast;
            }
            neighbor 10.255.14.224;
        }
    }
    ospf {
        traffic-engineering;
        area 0.0.0.0 {
            interface so-0/0/1.0;
        }
    }
    ldp {
        interface so-0/0/1.0;
```

```
    }
}
```

Configuring Policy Options on Router PE1

On Router PE1, you need to configure policies at the `[edit policy-options]` hierarchy level. These policies ensure that the CE routers in the Layer 3 VPN exchange routing information. In this example, Router CE1 in San Francisco exchanges routing information with Router CE2 in Chicago.

Configure the policy options on the PE1 router:

```
[edit]
policy-options {
    policy-statement vpn-import-VPN-A {
        term term1 {
            from {
                protocol bgp;
                community import-target-VPN-A;
            }
            then accept;
        }
        term term2 {
            then reject;
        }
    }
    policy-statement vpn-export-VPN-A {
        term term1 {
            from protocol ospf;
            then {
                community add export-target-VPN-A;
                accept;
            }
        }
        term term2 {
            then reject;
        }
    }
    community export-target-VPN-A members [target:10.255.14.216:11 domain-id:192.0.2.1:0];
    community import-target-VPN-A members target:10.255.14.224:31;
}
```

Configuring the Routing Instance on Router PE1

You need to configure a Layer 3 VPN routing instance on Router PE1. To indicate that the routing instance is for a Layer 3 VPN, add the `instance-type vrf` statement at the `[edit routing-instance` *`routing-instance-name`*`]` hierarchy level.

The `domain-id` statement is configured at the `[edit routing-instances routing-options protocols ospf]` hierarchy level. As shown in Figure 14 on page 170, the routing instance on Router PE2 must share the same domain ID as the corresponding routing instance on Router PE1 so that routes from Router CE1 to Router CE2 and vice versa are distributed as Type 3 LSAs. If you configure different OSPF domain IDs in the routing instances for Router PE1 and Router PE2, the routes from each CE router will be distributed as Type 5 LSAs.

Configure the routing instance on Router PE1:

```
[edit]
routing-instances {
    VPN-A-San-Francisco-Chicago {
        instance-type vrf;
        interface so-0/0/0.0;
        route-distinguisher 10.255.14.216:11;
        vrf-import vpn-import-VPN-A;
        vrf-export vpn-export-VPN-A;
        routing-options {
            router-id 10.255.14.216;
        }
        protocols {
            ospf {
                domain-id 192.0.2.1;
                export vpn-import-VPN-A;
                area 0.0.0.0 {
                    interface so-0/0/0.0;
                }
            }
        }
    }
}
```

Configuration Summary for Router PE1

Configure Interfaces

```
interfaces {
    so-0/0/0 {
        unit 0 {
            family inet {
                address 10.19.1.2/30;
            }
        }
    }
    so-0/0/1 {
        unit 0 {
            family inet {
                address 10.19.2.1/30;
            }
            family mpls;
        }
    }
}
```

Configure Routing Options

```
routing-options {
    router-id 10.255.14.216;
    autonomous-system 65069;
}
```

Configure Protocols

```
protocols {
    mpls {
        interface so-0/0/1.0;
    }
    bgp {
        group San-Francisco-Chicago {
            type internal;
            preference 10;
            local-address 10.255.14.216;
            family inet-vpn {
```

```
                unicast;
            }
            neighbor 10.255.14.224;
        }
    }
    ospf {
        traffic-engineering;
        area 0.0.0.0 {
            interface so-0/0/1.0;
        }
    }
    ldp {
        interface so-0/0/1.0;
    }
}
```

Configure VPN Policy

```
policy-options {
    policy-statement vpn-import-VPN-A {
        term term1 {
            from {
                protocol bgp;
                community import-target-VPN-A;
            }
            then accept;
        }
        term term2 {
            then reject;
        }
    }
    policy-statement vpn-export-VPN-A {
        term term1 {
            from protocol ospf;
            then {
                community add export-target-VPN-A;
                accept;
            }
        }
        term term2 {
            then reject;
        }
```

```
    }
    community export-target-VPN-A members [ target:10.255.14.216:11 domain-id:192.0.2.1:0 ];
    community import-target-VPN-A members target:10.255.14.224:31;
}
```

Routing Instance for Layer 3 VPN

```
routing-instances {
    VPN-A-San-Francisco-Chicago {
        instance-type vrf;
        interface so-0/0/0.0;
        route-distinguisher 10.255.14.216:11;
        vrf-import vpn-import-VPN-A;
        vrf-export vpn-export-VPN-A;
        routing-options {
            router-id 10.255.14.216;
        }
        protocols {
            ospf {
                domain-id 192.0.2.1;
                export vpn-import-VPN-A;
                area 0.0.0.0 {
                    interface so-0/0/0.0;
                }
            }
        }
    }
}
```

OSPFv2 Sham Links Overview

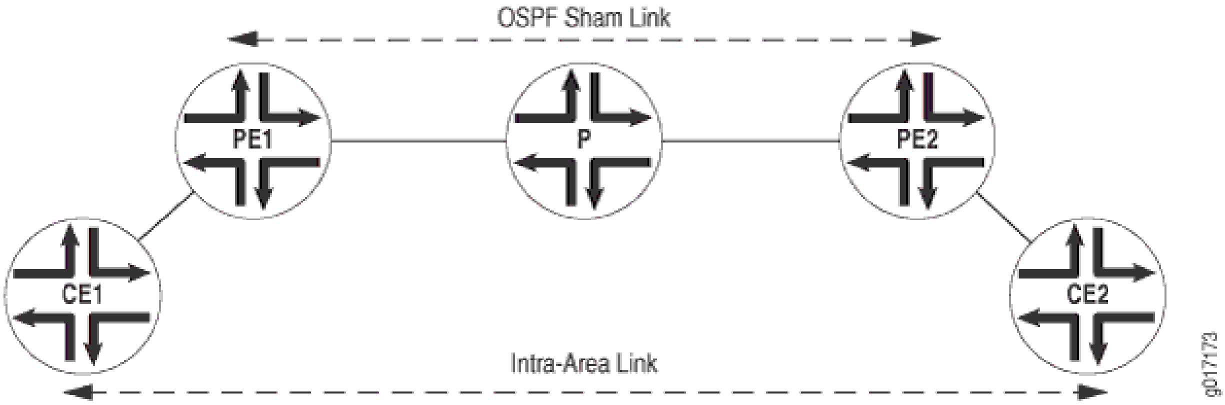
You can create an intra-area link or sham link between two provider edge (PE) routing devices so that the VPN backbone is preferred over the back-door link. A back-door link is a backup link that connects customer edge (CE) devices in case the VPN backbone is unavailable. When such a backup link is available and the CE devices are in the same OSPF area, the default behavior is to prefer this backup link over the VPN backbone. This is because the backup link is considered an intra-area link, while the VPN backbone is always considered an interarea link. Intra-area links are always preferred over interarea links.

The sham link is an unnumbered point-to-point intra-area link between PE devices. When the VPN backbone has a sham intra-area link, this sham link can be preferred over the backup link if the sham link has a lower OSPF metric than the backup link.

The sham link is advertised using Type 1 link-state advertisements (LSAs). Sham links are valid only for routing instances and OSPFv2.

Each sham link is identified by the combination of a local endpoint address and a remote endpoint address. Figure 15 on page 177 shows an OSPFv2 sham link. Router CE1 and Router CE2 are located in the same OSPFv2 area. These customer edge (CE) routing devices are linked together by a Layer 3 VPN over Router PE1 and Router PE2. In addition, Router CE1 and Router CE2 are connected by an intra-area link used as a backup.

Figure 15: OSPFv2 Sham Link



OSPFv2 treats the link through the Layer 3 VPN as an interarea link. By default, OSPFv2 prefers intra-area links to interarea links, so OSPFv2 selects the backup intra-area link as the active path. This is not acceptable in a configuration where the intra-area link is not the expected primary path for traffic between the CE routing devices. You can configure the metric for the sham link to ensure that the path over the Layer 3 VPN is preferred to a backup path over an intra-area link connecting the CE routing devices.

For the remote endpoint, you can configure the OSPFv2 interface as a demand circuit, configure IPsec authentication (you configure the actual IPsec authentication separately), and define the metric value.

You should configure an OSPFv2 sham link under the following circumstances:

- Two CE routing devices are linked together by a Layer 3 VPN.
- These CE routing devices are in the same OSPFv2 area.
- An intra-area link is configured between the two CE routing devices.

If there is no intra-area link between the CE routing devices, you do not need to configure an OSPFv2 sham link.

NOTE: In Junos OS Release 9.6 and later, an OSPFv2 sham link is installed in the routing table as a hidden route. Additionally, a BGP route is not exported to OSPFv2 if a corresponding OSPF sham link is available.

NOTE: In Junos OS Release 16.1 and later, OSPF sham-links are supported on default instances. The cost of the sham-link is dynamically set to the aigp-metric of the BGP route if no metric is configured on the sham-link by the user. If the aigp-metric is not present in the BGP route then the sham-link cost defaults to 1.

Example: Configuring OSPFv2 Sham Links

IN THIS SECTION

This example shows how to enable OSPFv2 sham links on a PE routing device.

Requirements

No special configuration beyond device initialization is required before configuring this example.

Overview

IN THIS SECTION

The sham link is an unnumbered point-to-point intra-area link and is advertised by means of a type 1 link-state advertisement (LSA). Sham links are valid only for routing instances and OSPFv2.

Each sham link is identified by a combination of the local endpoint address and a remote endpoint address and the OSPFv2 area to which it belongs. You manually configure the sham link between two PE devices, both of which are within the same VPN routing and forwarding (VRF) routing instance, and you specify the address for the local end point of the sham link. This address is used as the source for the sham link packets and is also used by the remote PE routing device as the sham link remote end point. You can also include the optional `metric` option to set a metric value for the remote end point. The metric value specifies the cost of using the link. Routes with lower total path metrics are preferred over those with higher path metrics.

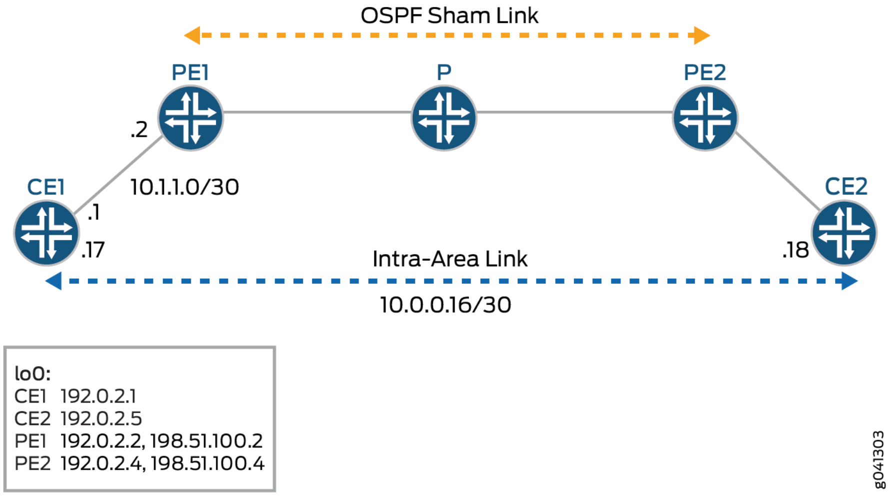
To enable OSPFv2 sham links on a PE routing device:

- Configure an extra loopback interface on the PE routing device.
- Configure the VRF routing instance that supports Layer 3 VPNs on the PE routing device, and associate the sham link with an existing OSPF area. The OSPFv2 sham link configuration is also included in the routing instance. You configure the sham link's local endpoint address, which is the loopback address of the local VPN, and the remote endpoint address, which is the loopback address of the remote VPN. In this example, the VRF routing instance is named red.

Figure 16 on page 179 shows an OSPFv2 sham link.

Topology

Figure 16: OSPFv2 Sham Link Example



The devices in the figure represent the following functions:

- CE1 and CE2 are the customer edge devices.
- PE1 and PE2 are the provider edge devices.

- P is the provider device.

"CLI Quick Configuration" on page 180 shows the configuration for all of the devices in Figure 16 on page 179. The section "Step-by-Step Procedure" on page 183 describes the steps on Device PE1.

Configuration

IN THIS SECTION

Procedure

CLI Quick Configuration

To quickly configure this example, copy the following commands, paste them into a text file, remove any line breaks, change any details necessary to match your network configuration, and then copy and paste the commands into the CLI at the `[edit]` hierarchy level.

CE1

```
set interfaces fe-1/2/0 unit 0 family inet address 10.1.1.1/30
set interfaces fe-1/2/0 unit 0 family mpls
set interfaces fe-1/2/1 unit 0 family inet address 10.0.0.17/30
set interfaces lo0 unit 0 family inet address 192.0.2.1/24
set protocols ospf area 0.0.0.0 interface fe-1/2/0.0
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set protocols ospf area 0.0.0.0 interface fe-1/2/1.0 metric 100
set policy-options policy-statement send-direct from protocol direct
set policy-options policy-statement send-direct then accept
set routing-options router-id 192.0.2.1
set routing-options autonomous-system 1
```

PE1

```
set interfaces fe-1/2/0 unit 0 family inet address 10.1.1.2/30
set interfaces fe-1/2/0 unit 0 family mpls
set interfaces fe-1/2/1 unit 0 family inet address 10.1.1.5/30
```

```
set interfaces fe-1/2/1 unit 0 family mpls
set interfaces lo0 unit 0 family inet address 192.0.2.2/24
set interfaces lo0 unit 1 family inet address 198.51.100.2/24
set protocols mpls interface fe-1/2/1.0
set protocols bgp group toR4 type internal
set protocols bgp group toR4 local-address 192.0.2.2
set protocols bgp group toR4 family inet-vpn unicast
set protocols bgp group toR4 neighbor 192.0.2.4
set protocols ospf area 0.0.0.0 interface fe-1/2/1.0
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set protocols ldp interface fe-1/2/1.0
set protocols ldp interface lo0.0
set policy-options policy-statement bgp-to-ospf term 1 from protocol bgp
set policy-options policy-statement bgp-to-ospf term 1 then accept
set policy-options policy-statement bgp-to-ospf term 2 then reject
set routing-instances red instance-type vrf
set routing-instances red interface fe-1/2/0.0
set routing-instances red interface lo0.1
set routing-instances red route-distinguisher 2:1
set routing-instances red vrf-target target:2:1
set routing-instances red protocols ospf export bgp-to-ospf
set routing-instances red protocols ospf sham-link local 198.51.100.2
set routing-instances red protocols ospf area 0.0.0.0 sham-link-remote 198.51.100.4 metric 10
set routing-instances red protocols ospf area 0.0.0.0 interface fe-1/2/0.0
set routing-instances red protocols ospf area 0.0.0.0 interface lo0.1
set routing-options router-id 192.0.2.2
set routing-options autonomous-system 2
```

P

```
set interfaces fe-1/2/0 unit 0 family inet address 10.1.1.6/30
set interfaces fe-1/2/0 unit 0 family mpls
set interfaces fe-1/2/1 unit 0 family inet address 10.1.1.9/30
set interfaces fe-1/2/1 unit 0 family mpls
set interfaces lo0 unit 3 family inet address 192.0.2.3/24
set protocols mpls interface all
set protocols ospf area 0.0.0.0 interface lo0.3 passive
set protocols ospf area 0.0.0.0 interface all
set protocols ldp interface all
set routing-options router-id 192.0.2.3
```

PE2

```
set interfaces fe-1/2/0 unit 0 family inet address 10.1.1.10/30
set interfaces fe-1/2/0 unit 0 family mpls
set interfaces fe-1/2/1 unit 0 family inet address 10.1.1.13/30
set interfaces fe-1/2/1 unit 0 family mpls
set interfaces lo0 unit 0 family inet address 192.0.2.4/32
set interfaces lo0 unit 1 family inet address 198.51.100.4/32
set protocols mpls interface fe-1/2/0.0
set protocols bgp group toR2 type internal
set protocols bgp group toR2 local-address 192.0.2.4
set protocols bgp group toR2 family inet-vpn unicast
set protocols bgp group toR2 neighbor 192.0.2.2
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set protocols ospf area 0.0.0.0 interface fe-1/2/0.0
set protocols ldp interface fe-1/2/0.0
set protocols ldp interface lo0.0
set policy-options policy-statement bgp-to-ospf term 1 from protocol bgp
set policy-options policy-statement bgp-to-ospf term 1 then accept
set policy-options policy-statement bgp-to-ospf term 2 then reject
set routing-instances red instance-type vrf
set routing-instances red interface fe-1/2/1.0
set routing-instances red interface lo0.1
set routing-instances red route-distinguisher 2:1
set routing-instances red vrf-target target:2:1
set routing-instances red protocols ospf export bgp-to-ospf
set routing-instances red protocols ospf sham-link local 198.51.100.4
set routing-instances red protocols ospf area 0.0.0.0 sham-link-remote 198.51.100.2 metric 10
set routing-instances red protocols ospf area 0.0.0.0 interface fe-1/2/1.0
set routing-instances red protocols ospf area 0.0.0.0 interface lo0.1
set routing-options router-id 192.0.2.4
set routing-options autonomous-system 2
```

CE2

```
set interfaces fe-1/2/0 unit 14 family inet address 10.1.1.14/30
set interfaces fe-1/2/0 unit 14 family mpls
set interfaces fe-1/2/0 unit 18 family inet address 10.0.0.18/30
set interfaces lo0 unit 5 family inet address 192.0.2.5/24
set protocols ospf area 0.0.0.0 interface fe-1/2/0.14
set protocols ospf area 0.0.0.0 interface lo0.5 passive
```

```
set protocols ospf area 0.0.0.0 interface fe-1/2/0.18
set policy-options policy-statement send-direct from protocol direct
set policy-options policy-statement send-direct then accept
set routing-options router-id 192.0.2.5
set routing-options autonomous-system 3
```

Step-by-Step Procedure

The following example requires you to navigate various levels in the configuration hierarchy. For information about navigating the CLI, see *Modifying the Junos OS Configuration* in CLI User Guide.

To configure OSPFv2 sham links on each PE device:

1. Configure the interfaces, including two loopback interfaces.

   ```
   [edit interfaces]
   user@PE1# set fe-1/2/0 unit 0 family inet address 10.1.1.2/30
   user@PE1# set fe-1/2/0 unit 0 family mpls
   user@PE1# set fe-1/2/1 unit 0 family inet address 10.1.1.5/30
   user@PE1# set fe-1/2/1 unit 0 family mpls
   user@PE1# set lo0 unit 0 family inet address 192.0.2.2/24
   user@PE1# set lo0 unit 1 family inet address 198.51.100.2/24
   ```

2. Configure MPLS on the core-facing interface.

   ```
   [edit protocols mpls]
   user@PE1# set interface fe-1/2/1.0
   ```

3. Configure internal BGP (IBGP).

   ```
   [edit ]
   user@PE1# set protocols bgp group toR4 type internal
   user@PE1# set protocols bgp group toR4 local-address 192.0.2.2
   user@PE1# set protocols bgp group toR4 family inet-vpn unicast
   user@PE1# set protocols bgp group toR4 neighbor 192.0.2.4
   ```

4. Configure OSPF on the core-facing interface and on the loopback interface that is being used in the main instance.

```
[edit protocols ospf area 0.0.0.0]
user@PE1# set interface fe-1/2/1.0
user@PE1# set interface lo0.0 passive
```

5. Configure LDP or RSVP on the core-facing interface and on the loopback interface that is being used in the main instance.

```
[edit protocols ldp]
user@PE1# set interface fe-1/2/1.0
user@PE1# set interface lo0.0
```

6. Configure a routing policy for use in the routing instance.

```
[edit policy-options policy-statement bgp-to-ospf]
user@PE1# set term 1 from protocol bgp
user@PE1# set term 1 then accept
user@PE1# set term 2 then reject
```

7. Configure the routing instance.

```
[edit routing-instances red]
user@PE1# set instance-type vrf
user@PE1# set interface fe-1/2/0.0
user@PE1# set route-distinguisher 2:1
user@PE1# set vrf-target target:2:1
user@PE1# set protocols ospf export bgp-to-ospf
user@PE1# set protocols ospf area 0.0.0.0 interface fe-1/2/0.0
```

8. Configure the OSPFv2 sham link.

   Include the extra loopback interface in the routing instance and also in the OSPF configuration.

Notice that the metric on the sham-link interface is set to 10. On Device CE1's backup OSPF link, the metric is set to 100. This causes the sham link to be the preferred link.

```
[edit routing-instances red]
user@PE1# set interface lo0.1
user@PE1# set protocols ospf sham-link local 198.51.100.2
user@PE1# set protocols ospf area 0.0.0.0 sham-link-remote 198.51.100.4 metric 10
user@PE1# set protocols ospf area 0.0.0.0 interface lo0.1
```

9. Configure the autonomous system (AS) number and the router ID.

```
[edit routing-options]
user@PE1# set router-id 192.0.2.2
user@PE1# set autonomous-system 2
```

10. If you are done configuring the device, commit the configuration.

```
[edit]
user@R1# commit
```

Results

Confirm your configuration by entering the `show interfaces` and the `show routing-instances` commands. If the output does not display the intended configuration, repeat the instructions in this example to correct the configuration.

Output for PE1:

```
user@PE1# show interfaces
fe-1/2/0 {
    unit 0{
        family inet {
            address 10.1.1.2/30;
        }
        family mpls;
    }
}
fe-1/2/1 {
    unit 0 {
```

```
        family inet {
            address 10.1.1.5/30;
        }
        family mpls;
    }
}
lo0 {
    unit 0 {
        family inet {
            address 192.0.2.2/24;
        }
    }
    unit 1 {
        family inet {
            address 198.51.100.2/24;
        }
    }
}
```

```
user@PE1# show protocols
mpls {
    interface fe-1/2/1.0;
}
bgp {
    group toR4 {
        type internal;
        local-address 192.0.2.2;
        family inet-vpn {
            unicast;
        }
        neighbor 192.0.2.4;
    }
}
ospf {
    area 0.0.0.0 {
        interface fe-1/2/1.0;
        interface lo0.0 {
            passive;
        }
    }
}
```

```
ldp {
    interface fe-1/2/1.0;
    interface lo0.0;
}
```

```
user@PE1# show policy-options
policy-statement bgp-to-ospf {
    term 1 {
        from protocol bgp;
        then accept;
    }
    term 2 {
        then reject;
    }
}
```

```
user@PE1# show routing-instances
red {
    instance-type vrf;
    interface fe-1/2/0.0;
    interface lo0.1;
    route-distinguisher 2:1;
    vrf-target target:2:1;
    protocols {
        ospf {
            export bgp-to-ospf;
            sham-link local 198.51.100.2;
            area 0.0.0.0 {
                sham-link-remote 198.51.100.4 metric 10;
                interface fe-1/2/0.0;
                interface lo0.1;
            }
        }
```

```
    }
}
```

```
user@PE1# show routing-options
router-id 192.0.2.2;
autonomous-system 2;
```

Verification

IN THIS SECTION

Confirm that the configuration is working properly.

Verifying the Sham Link Interfaces

Purpose

Verify the sham link interface. The sham link is treated as an interface in OSPFv2, with the named displayed as `shamlink.<unique identifier>`, where the unique identifier is a number. For example, `shamlink.0`. The sham link appears as a point-to-point interface.

Action

From operational mode, enter the `show ospf interface instance` *`instance-name`* command.

```
user@PE1> show ospf interface instance red
Interface           State   Area            DR ID           BDR ID          Nbrs
lo0.1              DR      0.0.0.0         198.51.100.2         0.0.0.0            0
```

```
fe-1/2/0.0          PtToPt  0.0.0.0         0.0.0.0         0.0.0.0            1
shamlink.0          PtToPt  0.0.0.0         0.0.0.0         0.0.0.0            1
```

Verifying the Local and Remote End Points of the Sham Link

Purpose

Verify the local and remote end points of the sham link. The MTU for the sham link interface is always zero.

Action

From operational mode, enter the `show ospf interface instance` *`instance-name`* `detail` command.

```
user@PE1> show ospf interface shamlink.0 instance red
Interface           State   Area            DR ID           BDR ID          Nbrs
shamlink.0          PtToPt  0.0.0.0         0.0.0.0         0.0.0.0            1
  Type: P2P, Address: 0.0.0.0, Mask: 0.0.0.0, MTU: 0, Cost: 10
  Local: 198.51.100.2, Remote: 198.51.100.4
  Adj count: 1
  Hello: 10, Dead: 40, ReXmit: 5, Not Stub
  Auth type: None
  Protection type: None, No eligible backup
  Topology default (ID 0) -> Cost: 10
```

Verifying the Sham Link Adjacencies

Purpose

Verify the adjacencies between the configured sham links.

Action

From operational mode, enter the `show ospf neighbor instance` *`instance-name`* command.

```
user@PE1> show ospf neighbor instance red
Address          Interface              State     ID               Pri  Dead
```

```
10.1.1.1         fe-1/2/0.0             Full      192.0.2.1          128    35
198.51.100.4          shamlink.0             Full      198.51.100.4            0    31
```

Verifying the Link-State Advertisement

Purpose

Verify that the router LSA originated by the instance carries the sham link adjacency as an unnumbered point-to-point link. The link data for sham links is a number ranging from 0x80010000 through 0x8001ffff.

Action

From operational mode, enter the `show ospf database instance` *`instance-name`* command.

```
user@PE1> show ospf database instance red

    OSPF database, Area 0.0.0.0
 Type       ID               Adv Rtr           Seq      Age  Opt  Cksum  Len
Router   192.0.2.1          192.0.2.1          0x80000009  1803  0x22 0x6ec7  72
Router   192.0.2.5          192.0.2.5          0x80000007    70  0x22 0x2746  72
Router  *198.51.100.2          198.51.100.2          0x80000006    55  0x22 0xda6b  60
Router   198.51.100.4          198.51.100.4          0x80000005    63  0x22 0xb19   60
Network  10.0.0.18        192.0.2.5          0x80000002    70  0x22 0x9a71  32
    OSPF AS SCOPE link state database
 Type       ID               Adv Rtr           Seq      Age  Opt  Cksum  Len
Extern   198.51.100.2          198.51.100.4          0x80000002    72  0xa2 0x343   36
Extern  *198.51.100.4          198.51.100.2          0x80000002    71  0xa2 0xe263  36
```

Verifying the Path Selection

Purpose

Verify that the Layer 3 VPN path is used instead of the backup path.

Action

From operational mode, enter the `traceroute` command from Device CE1 to Device CE2.

```
user@CE1> traceroute 192.0.2.5

traceroute to 192.0.2.5 (192.0.2.5), 30 hops max, 40 byte packets
 1  10.1.1.2 (10.1.1.2)  1.930 ms  1.664 ms  1.643 ms
 2  * * *
 3  10.1.1.10 (10.1.1.10)  2.485 ms  1.435 ms  1.422 ms
     MPLS Label=299808 CoS=0 TTL=1 S=1
 4  192.0.2.5 (192.0.2.5)  1.347 ms  1.362 ms  1.329 ms
```

Meaning

The traceroute operation shows that the Layer 3 VPN is the preferred path. If you were to remove the sham link or if you were to modify the OSPF metric to prefer that backup path, the traceroute would show that the backup path is preferred.

Configuring EBGP Multihop Sessions Between PE and CE Routers in Layer 3 VPNs

You can configure an EBGP or IBGP multihop session between the PE and CE routers of a Layer 3 VPN. This allows you to have one or more routers between the PE and CE routers. Using IBGP between PE and CE routers does not require the configuration of any additional statements. However, using EBGP between the PE and CE routers requires the configuration of the `multihop` statement.

To configure an external BGP multihop session for the connection between the PE and CE routers, include the `multihop` statement on the PE router. To help prevent routing loops, you have to configure a time-to-live (TTL) value for the multihop session:

```
multihop ttl-value;
```

For the list of hierarchy levels at which you can configure this statement, see the summary section for this statement.

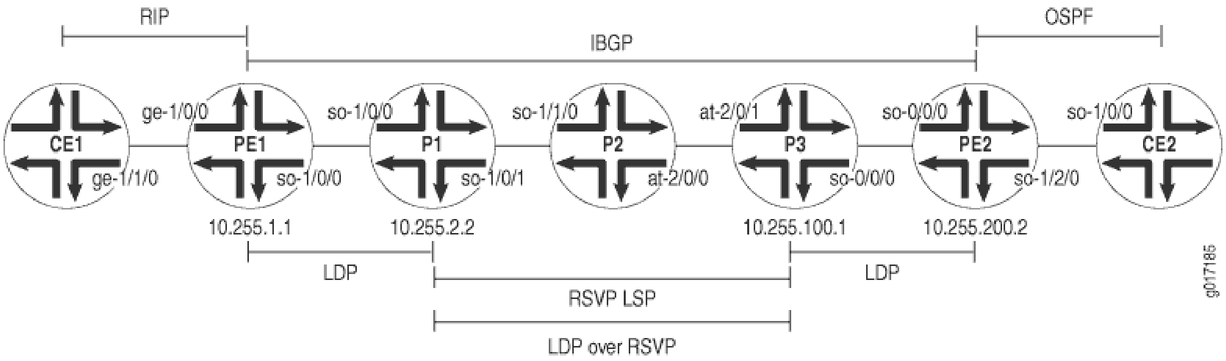
Configuring an LDP-over-RSVP VPN Topology

IN THIS SECTION

This example shows how to set up a VPN topology in which LDP packets are tunneled over an RSVP LSP. This configuration consists of the following components (see Figure 17 on page 192):

- One VPN (VPN-A)
- Two PE routers
- LDP as the signaling protocol between the PE routers and their adjacent P routers
- An RSVP LSP between two of the P routers over which LDP is tunneled

Figure 17: Example of an LDP-over-RSVP VPN Topology



The following steps describe how this topology is established and how packets are sent from CE Router CE2 to CE Router CE1:

1. The P routers P1 and P3 establish RSVP LSPs between each other and install their loopback addresses in their inet.3 routing tables.

2. PE Router PE1 establishes an LDP session with Router P1 over interface `so-1/0/0.0`.

3. Router P1 establishes an LDP session with Router P3's loopback address, which is reachable using the RSVP LSP.

4. Router P1 sends its label bindings, which include a label to reach Router PE1, to Router P3. These label bindings allow Router P3 to direct LDP packets to Router PE1.

5. Router P3 establishes an LDP session with Router PE2 over interface `so0-0/0/0.0` and establishes an LDP session with Router P1's loopback address.

6. Router P3 sends its label bindings, which include a label to reach Router PE2, to Router P1. These label bindings allow Router P1 to direct LDP packets to Router PE2's loopback address.

7. Routers PE1 and PE2 establish IBGP sessions with each other.

8. When Router PE1 announces to Router PE2 routes that it learned from Router CE1, it includes its VPN label. (The PE router creates the VPN label and binds it to the interface between the PE and CE routers.) Similarly, when Router PE2 announces routes that it learned from Router CE2, it sends its VPN label to Router PE1.

When Router PE2 wants to forward a packet to Router CE1, it pushes two labels onto the packet's label stack: first the VPN label that is bound to the interface between Router PE1 and Router CE1, then the LDP label used to reach Router PE1. Then it forwards the packets to Router P3 over interface `so-0/0/1.0`.

1. When Router P3 receives the packets from Router PE2, it swaps the LDP label that is on top of the stack (according to its LDP database) and also pushes an RSVP label onto the top of the stack so that the packet can now be switched by the RSVP LSP. At this point, there are three labels on the stack: the inner (bottom) label is the VPN label, the middle is the LDP label, and the outer (top) is the RSVP label.

2. Router P2 receives the packet and switches it to Router P1 by swapping the RSVP label. In this topology, because Router P2 is the penultimate-hop router in the LSP, it pops the RSVP label and forwards the packet over interface `so-1/1/0.0` to Router P1. At this point, there are two labels on the stack: The inner label is the VPN label, and the outer one is the LDP label.

3. When Router P1 receives the packet, it pops the outer label (the LDP label) and forwards the packet to Router PE1 using interface `so-1/0/0.0`. In this topology, Router PE1 is the egress LDP router, so Router P1 pops the LDP label instead of swapping it with another label. At this point, there is only one label on the stack, the VPN label.

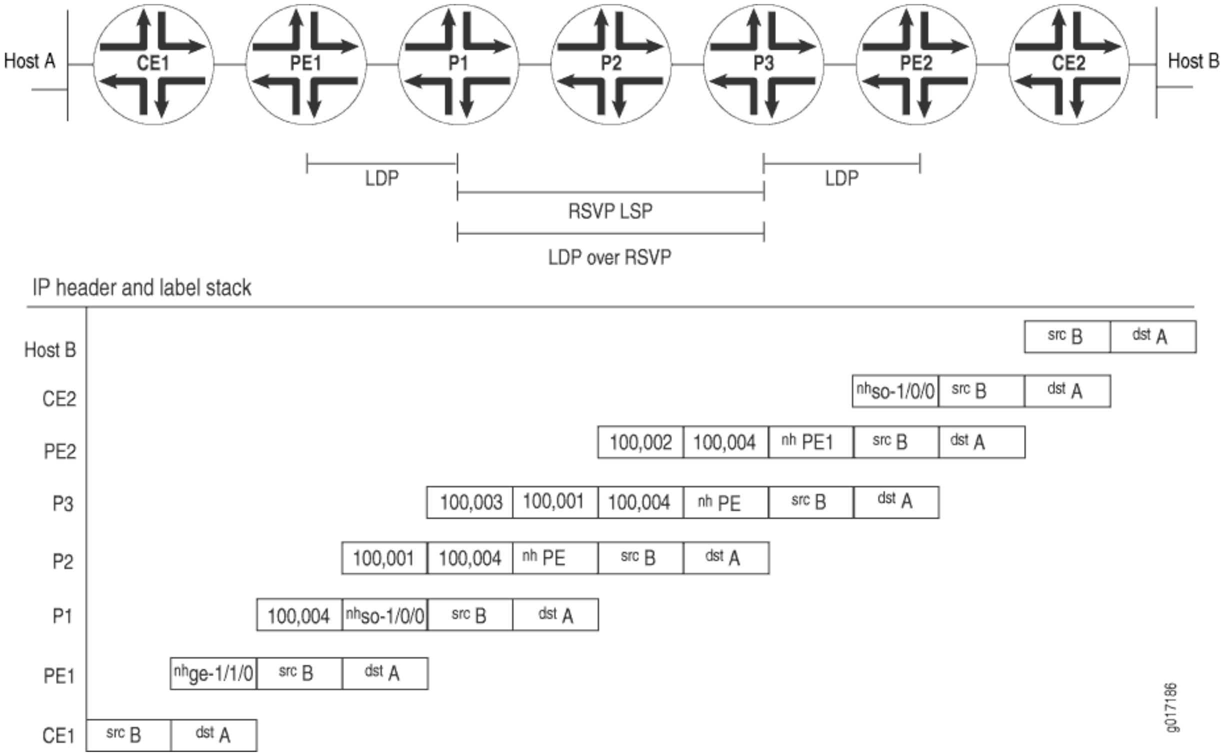
4. When Router PE1 receives the packet, it pops the VPN label and forwards the packet as an IPv4 packet to Router CE1 over interface `ge-1/1/0.0`.

A similar set of operations occurs for packets sent from Router CE1 that are destined for Router CE2.

The following list explains how, for packets being sent from Router CE2 to Router CE1, the LDP, RSVP, and VPN labels are announced by the various routers. These steps include examples of label values (illustrated in Figure 18 on page 195).

- LDP labels
    - Router PE1 announces LDP label 3 for itself to Router P1.
    - Router P1 announces LDP label 100,001 for Router PE1 to Router P3.
    - Router P3 announces LDP label 100,002 for Router PE1 to Router PE2.
- RSVP labels
    - Router P1 announces RSVP label 3 to Router P2.
    - Router P2 announces RSVP label 100,003 to Router P3.
- VPN label
    - Router PE1 announces VPN label 100,004 to Router PE2 for the route from Router CE1 to Router CE2.

Figure 18: Label Pushing and Popping



For a packet sent from Host B in Figure 18 on page 195 to Host A, the packet headers and labels change as the packet travels to its destination:

1. The packet that originates from Host B has a source address of B and a destination address of A in its header.
2. Router CE2 adds to the packet a next hop of interface `so-1/0/0`.
3. Router PE2 swaps out the next hop of interface `so-1/0/0` and replaces it with a next hop of PE1. It also adds two labels for reaching Router PE1, first the VPN label (100,004), then the LDP label (100,002). The VPN label is thus the inner (bottom) label on the stack, and the LDP label is the outer label.
4. Router P3 swaps out the LDP label added by Router PE2 (100,002) and replaces it with its LDP label for reaching Router PE1 (100,001). It also adds the RSVP label for reaching Router P2 (100,003).
5. Router P2 removes the RSVP label (100,003) because it is the penultimate hop in the MPLS LSP.
6. Router P1 removes the LDP label (100,001) because it is the penultimate LDP router. It also swaps out the next hop of PE1 and replaces it with the next-hop interface, `so-1/0/0`.
7. Router PE1 removes the VPN label (100,004). It also swaps out the next-hop interface of `so-1/0/0` and replaces it with its next-hop interface, `ge-1/1/0`.

8. Router CE1 removes the next-hop interface of `ge-1/1/0`, and the packet header now contains just a source address of B and a destination address of A.

The final section in this example consolidates the statements needed to configure VPN functionality on each of the service P routers shown in Figure 17 on page 192.

> **NOTE**: In this example, a private AS number is used for the route distinguisher and the route target. This number is used for illustration only. When you are configuring VPNs, you should use an assigned AS number.

The following sections explain how to configure the VPN functionality on the PE and P routers. The CE routers do not have any information about the VPN, so you configure them normally.

Enabling an IGP on the PE and P Routers

To allow the PE and P routers to exchange routing information among themselves, you must configure an IGP on all these routers or you must configure static routes. You configure the IGP on the primary instance of the routing protocol process (rpd) (that is, at the `[edit protocols]` hierarchy level), not within the VPN routing instance (that is, not at the `[edit routing-instances]` hierarchy level).

You configure the IGP in the standard way. This configuration example does not include this portion of the configuration.

Enabling LDP on the PE and P Routers

In this configuration example, the LDP is the signaling protocol between the PE routers. For the VPN to function, you must configure LDP on the two PE routers and on the P routers that are connected to the PE routers. You need to configure LDP only on the interfaces in the core of the service provider's network; that is, between the PE and P routers and between the P routers. You do not need to configure LDP on the interface between the PE and CE routers.

In this configuration example, you configure LDP on the P routers' loopback interfaces because these are the interfaces on which the MPLS LSP is configured.

On the PE routers, you must also configure `family inet` when you configure the logical interface.

On Router PE1, configure LDP:

```
[edit protocols]
ldp {
    interface so-1/0/0.0;
}
[edit interfaces]
```

```
so-1/0/0 {
    unit 0 {
        family mpls;
    }
}
```

On Router PE2, configure LDP:

```
[edit protocols]
ldp {
    interface so-0/0/0.0;
}
[edit interfaces]
so-0/0/1 {
    unit 0 {
        family mpls;
    }
}
```

On Router P1, configure LDP:

```
[edit protocols]
ldp {
    interface so-1/0/0.0;
    interface lo0;
}
```

On Router P3, configure LDP:

```
[edit protocols]
ldp {
    interface lo0;
    interface so-0/0/0.0;
}
```

On Router P2, although you do not need to configure LDP, you can optionally configure it to provide a fallback LDP path in case the RSVP LSP becomes nonoperational:

```
[edit protocols]
ldp {
```

```
    interface so-1/1/0.0;
    interface at-2/0/0.0;
}
```

Enabling RSVP and MPLS on the P Router

On the P Router P2 you must configure RSVP and MPLS because this router exists on the MPLS LSP path between the P Routers P1 and P3:

```
[edit]
protocols {
    rsvp {
        interface so-1/1/0.0;
        interface at-2/0/0.0;
    }
    mpls {
        interface so-1/1/0.0;
        interface at-2/0/0.0;
    }
}
```

Configuring the MPLS LSP Tunnel Between the P Routers

In this configuration example, LDP is tunneled over an RSVP LSP. Therefore, in addition to configuring RSVP, you must enable traffic engineering support in an IGP, and you must create an MPLS LSP to tunnel the LDP traffic.

On Router P1, enable RSVP and configure one end of the MPLS LSP tunnel. In this example, traffic engineering support is enabled for OSPF, and you configure MPLS on the interfaces to the LSP and to Router PE1. In the `to` statement, you specify the loopback address of Router P3.

```
[edit]
protocols {
    rsvp {
        interface so-1/0/1.0;
    }
    mpls {
        label-switched-path P1-to-P3 {
            to 10.255.100.1;
            ldp-tunneling;
```

```
        }
        interface so-1/0/0.0;
        interface so-1/0/1.0;
    }
    ospf {
        traffic-engineering;
        area 0.0.0.0 {
            interface so-1/0/0.0;
            interface so-1/0/1.0;
        }
    }
}
```

On Router P3, enable RSVP and configure the other end of the MPLS LSP tunnel. Again, traffic engineering support is enabled for OSPF, and you configure MPLS on the interfaces to the LSP and to Router PE2. In the `to` statement, you specify the loopback address of Router P1.

```
[edit]
protocols {
    rsvp {
        interface at-2/0/1.0;
    }
    mpls {
        label-switched-path P3-to-P1 {
            to 10.255.2.2;
            ldp-tunneling;
        }
        interface at-2/0/1.0;
        interface so-0/0/0.0;
    }
    ospf {
        traffic-engineering;
        area 0.0.0.0 {
            interface at-2/0/1.0;
            interface so-0/0/0.0;
        }
    }
}
```

Configuring IBGP on the PE Routers

On the PE routers, configure an IBGP session with the following properties:

- VPN family—To indicate that the IBGP session is for the VPN, include the `family inet-vpn` statement.
- Loopback address—Include the `local-address` statement, specifying the local PE router's loopback address. The IBGP session for VPNs runs through the loopback address. You must also configure the `lo0` interface at the `[edit interfaces]` hierarchy level. The example does not include this part of the router's configuration.
- Neighbor address—Include the `neighbor` statement, specifying the IP address of the neighboring PE router, which is its loopback (`lo0`) address.

On Router PE1, configure IBGP:

```
[edit]
protocols {
    bgp {
        group PE1-to-PE2 {
            type internal;
            local-address 10.255.1.1;
            family inet-vpn {
                unicast;
            }
            neighbor 10.255.200.2;
        }
    }
}
```

On Router PE2, configure IBGP:

```
[edit]
protocols {
    bgp {
        group PE2-to-PE1 {
            type internal;
            local-address 10.255.200.2;
            family inet-vpn {
                unicast;
            }
            neighbor 10.255.1.1;
        }
```

```
    }
}
```

Configuring Routing Instances for VPNs on the PE Routers

Both PE routers service VPN-A, so you must configure one routing instance on each router for the VPN in which you define the following:

- Route distinguisher, which must be unique for each routing instance on the PE router. It is used to distinguish the addresses in one VPN from those in another VPN.
- Instance type of `vrf`, which creates the VRF table on the PE router.
- Interfaces connected to the CE routers.
- VRF import and export policies, which must be the same on each PE router that services the same VPN. Unless the import policy contains only a `then reject` statement, it must include reference to a community. Otherwise, when you try to commit the configuration, the commit fails.

> **NOTE**: In this example, a private AS number is used for the route distinguisher. This number is used for illustration only. When you are configuring VPNs, you should use an assigned AS number.

- Routing between the PE and CE routers, which is required for the PE router to distribute VPN-related routes to and from connected CE routers. You can configure a routing protocol—BGP, OSPF, or RIP—or you can configure static routing.

On Router PE1, configure the following routing instance for VPN-A. In this example, Router PE1 uses RIP to distribute routes to and from the CE router to which it is connected.

```
[edit]
routing-instance {
    VPN-A {
        instance-type vrf;
        interface ge-1/0/0.0;
        route-distinguisher 65535:0;
        vrf-import VPN-A-import;
        vrf-export VPN-A-export;
        protocols {
            rip {
                group PE1-to-CE1 {
                    neighbor ge-1/0/0.0;
```

```
                }
            }
        }
    }
}
```

On Router PE2, configure the following routing instance for VPN-A. In this example, Router PE2 uses OSPF to distribute routes to and from the CE router to which it is connected.

```
[edit]
routing-instance {
    VPN-A {
        instance-type vrf;
        interface so-1/2/0.0;
        route-distinguisher 65535:1;
        vrf-import VPN-A-import;
        vrf-export VPN-A-export;
        protocols {
            ospf {
                area 0.0.0.0 {
                    interface so-1/2/0.0;
                }
            }
        }
    }
}
```

Configuring VPN Policy on the PE Routers

You must configure VPN import and export policies on each of the PE routers so that they install the appropriate routes in their VRF tables, which they use to forward packets within a VPN. For VPN-A, the VRF table is VPN-A.inet.0.

In the VPN policy, you also configure VPN target communities.

NOTE: In this example, a private AS number is used for the route target. This number is used for illustration only. When you are configuring VPNs, you should use an assigned AS number.

On Router PE1, configure the following VPN import and export policies:

NOTE: The policy qualifiers shown in this example are only those needed for the VPN to function. You can configure additional qualifiers, as needed, to any policies that you configure.

```
[edit]
policy-options {
    policy-statement VPN-A-import {
        term a {
            from {
                protocol bgp;
                community VPN-A;
            }
            then accept;
        }
        term b {
            then reject;
        }
    }
    policy-statement VPN-A-export {
        term a {
            from protocol rip;
            then {
                community add VPN-A;
                accept;
            }
        }
        term b {
            then reject;
        }
    }
    community VPN-A members target:65535:00;
}
```

On Router PE2, configure the following VPN import and export policies:

```
[edit]
policy-options {
    policy-statement VPN-A-import {
        term a {
```

```
            from {
                protocol bgp;
                community VPN-A;
            }
            then accept;
        }
        term b {
            then reject;
        }
    }
    policy-statement VPN-A-export {
        term a {
            from protocol ospf;
            then {
                community add VPN-A;
                accept;
            }
        }
        term b {
            then reject;
        }
    }
    community VPN-A members target:65535:00;
}
```

To apply the VPN policies on the routers, include the `vrf-export` and `vrf-import` statements when you configure the routing instance on the PE routers. The VRF import and export policies handle the route distribution across the IBGP session running between the PE routers.

LDP-over-RSVP VPN Configuration Summarized by Router

Router PE1

Routing Instance for VPN-A

```
routing-instance {
    VPN-A {
        instance-type vrf;
        interface ge-1/0/0.0;
        route-distinguisher 65535:0;
        vrf-import VPN-A-import;
        vrf-export VPN-A-export;
```

```
    }
}
```

Instance Routing Protocol

```
protocols {
    rip {
        group PE1-to-CE1 {
            neighbor ge-1/0/0.0;
        }
    }
}
```

Interfaces

```
interfaces {
    so-1/0/0 {
        unit 0 {
            family mpls;
        }
    }
    ge-1/0/0 {
        unit 0;
    }
}
```

Primary Protocol Instance

```
protocols {
}
```

Enable LDP

```
ldp {
    interface so-1/0/0.0;
}
```

Enable MPLS

```
mpls {
    interface so-1/0/0.0;
    interface ge-1/0/0.0;
}
```

Configure IBGP

```
bgp {
    group PE1-to-PE2 {
        type internal;
        local-address 10.255.1.1;
        family inet-vpn {
            unicast;
        }
        neighbor 10.255.100.1;
    }
}
```

Configure VPN Policy

```
policy-options {
    policy-statement VPN-A-import {
        term a {
            from {
                protocol bgp;
                community VPN-A;
            }
            then accept;
        }
        term b {
            then reject;
        }
    }
    policy-statement VPN-A-export {
        term a {
            from protocol rip;
            then {
                community add VPN-A;
                accept;
```

```
            }
        }
        term b {
            then reject;
        }
    }
    community VPN-A members target:65535:00;
}
```

Router P1

Primary Protocol Instance

```
protocols {
}
```

Enable RSVP

```
rsvp {
    interface so-1/0/1.0;
}
```

Enable LDP

```
ldp {
    interface so-1/0/0.0;
    interface lo0.0;
}
```

Enable MPLS

```
mpls {
    label-switched-path P1-to-P3 {
        to 10.255.100.1;
        ldp-tunneling;
    }
    interface so-1/0/0.0;
```

```
    interface so-1/0/1.0;
}
```

Configure OSPF for Traffic Engineering Support

```
ospf {
    traffic-engineering;
    area 0.0.0.0 {
        interface so-1/0/0.0;
        interface so-1/0/1.0;
    }
}
```

Router P2

Primary Protocol Instance

```
protocols {
}
```

Enable RSVP

```
rsvp {
    interface so-1/1/0.0;
    interface at-2/0/0.0;
}
```

Enable MPLS

```
mpls {
    interface so-1/1/0.0;
    interface at-2/0/0.0;
}
```

Router P3

Primary Protocol Instance

```
protocols {
}
```

Enable RSVP

```
rsvp {
    interface at-2/0/1.0;
}
```

Enable LDP

```
ldp {
    interface so-0/0/0.0;
    interface lo0.0;
}
```

Enable MPLS

```
mpls {
    label-switched-path P3-to-P1 {
        to 10.255.2.2;
        ldp-tunneling;
    }
    interface at-2/0/1.0;
    interface so-0/0/0.0;
}
```

Configure OSPF for Traffic Engineering Support

```
ospf {
    traffic-engineering;
    area 0.0.0.0 {
        interface at-2/0/1.0;
        interface at-2/0/1.0;
```

```
    }
}
```

Router PE2

Routing Instance for VPN-A

```
routing-instance {
    VPN-A {
        instance-type vrf;
        interface so-1/2/0.0;
        route-distinguisher 65535:1;
        vrf-import VPN-A-import;
        vrf-export VPN-A-export;
    }
}
```

Instance Routing Protocol

```
protocols {
    ospf {
        area 0.0.0.0 {
            interface so-1/2/0.0;
        }
    }
}
```

Interfaces

```
interfaces {
    so-0/0/0 {
        unit 0 {
            family mpls;
        }
    }
    so-1/2/0 {
        unit 0;
    }
}
```

Primary Protocol Instance

```
protocols {
}
```

Enable LDP

```
ldp {
    interface so-0/0/0.0;
}
```

Enable MPLS

```
mpls {
    interface so-0/0/0.0;
    interface so-1/2/0.0;
}
```

Configure IBGP

```
bgp {
    group PE2-to-PE1 {
        type internal;
        local-address 10.255.200.2;
        family inet-vpn {
            unicast;
        }
        neighbor 10.255.1.1;
    }
}
```

Configure VPN Policy

```
policy-options {
    policy-statement VPN-A-import {
        term a {
            from {
                protocol bgp;
                community VPN-A;
```

```
            }
            then accept;
        }
        term b {
            then reject;
        }
    }
    policy-statement VPN-A-export {
        term a {
            from protocol ospf;
            then {
                community add VPN-A;
                accept;
            }
        }
        term b {
            then reject;
        }
    }
    community VPN-A members target:65535:01;
}
```

Configuring an Application-Based Layer 3 VPN Topology

IN THIS SECTION

This example illustrates an application-based mechanism for forwarding traffic into a Layer 3 VPN. Typically, one or more interfaces are associated with, or bound to, a VPN by including them in the configuration of the VPN routing instance. By binding the interface to the VPN, the VPN's VRF table is used to make forwarding decisions for any incoming traffic on that interface. Binding the interface also includes the interface local routes in the VRF table, which provides next-hop resolution for VRF routes.

In this example, a firewall filter is used to define which incoming traffic on an interface is forwarded by means of the standard routing table, inet.0, and which incoming traffic is forwarded by means of the VRF table. You can expand this example such that incoming traffic on an interface can be redirected to one or more VPNs. For example, you can define a configuration to support a VPN that forwards traffic based on source address, that forwards Hypertext Transfer Protocol (HTTP) traffic, or that forwards only streaming media.

For this configuration to work, the following conditions must be true:

- The interfaces that use filter-based forwarding must not be bound to the VPN.
- Static routing must be used as the means of routing.
- You must define an interface routing table group that is shared among inet.0 and the VRF tables to provide local routes to the VRF table.

This example consists of two client hosts (Client D and Client E) that are in two different VPNs and that want to send traffic both within the VPN and to the Internet. The paths are defined as follows:

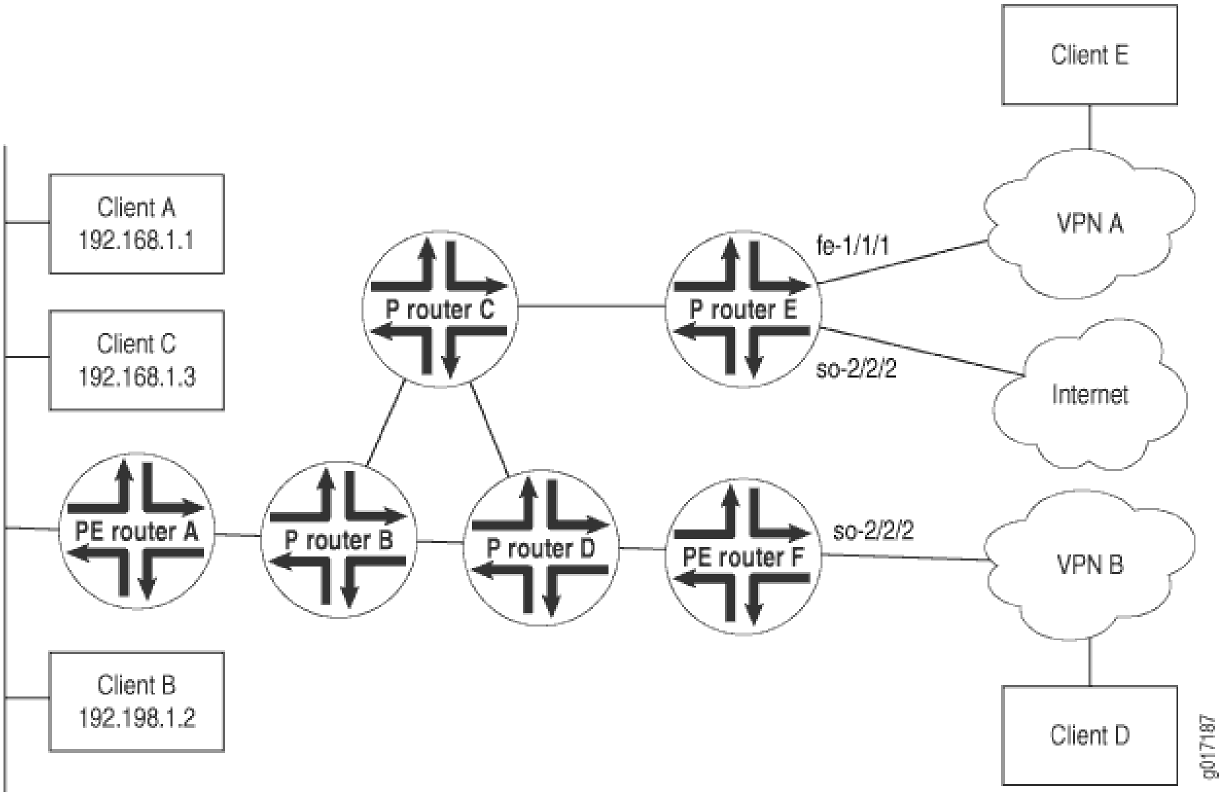
- Client A sends traffic to Client E over VPN A with a return path that also uses VPN A (using the VPN's VRF table).
- Client B sends traffic to Client D over VPN B with a return path that uses standard destination-based routing (using the inet.0 routing table).
- Clients B and C send traffic to the Internet using standard routing (using the inet.0 routing table), with a return path that also uses standard routing.

This example illustrates that there are a large variety of options in configuring an application-based Layer 3 VPN topology. This flexibility has application in many network implementations that require specific traffic to be forwarded in a constrained routing environment.

This configuration example shows only the portions of the configuration for the filter-based forwarding, routing instances, and policy. It does not illustrate how to configure a Layer 3 VPN.

Figure 19 on page 214 illustrates the network topology used in this example.

Figure 19: Application-Based Layer 3 VPN Example Configuration



Configuration on Router A

On Router A, you configure the interface to Clients A, B, and C. The configuration evaluates incoming traffic to determine whether it is to be forwarded by means of VPN or standard destination-based routing.

First, you apply an inbound filter and configure the interface:

```
[edit]
interfaces {
    fe-1/1/0 {
        unit 0 {
            family inet {
                filter {
                    input fbf-vrf;
                }
                address 192.168.1.1/24;
            }
        }
```

```
    }
}
```

Because the interfaces that use filter-based forwarding must not be bound to a VPN, you must configure an alternate method to provide next-hop routes to the VRF table. You do this by defining an interface routing table group and sharing this group among all the routing tables:

```
[edit]
routing-options {
    interface-routes {
        rib-group inet if-rib;
    }
    rib-groups {
        if-rib {
            import-rib [ inet.0 vpn-A.inet.0 vpn-B.inet.0 ];
        }
    }
}
```

You apply the following filter to incoming traffic on interface `fe-1/1/0.0`. The first term matches traffic from Client A and forwards it to the routing instance for VPN A. The second term matches traffic from Client B that is destined for Client D and forwards it to the routing instance for VPN B. The third term matches all other traffic, which is forwarded normally by means of destination-based forwarding according to the routes in inet.0.

```
[edit firewall family family-name]
filter fbf-vrf {
    term vpnA {
        from {
            source-address {
                192.168.1.1/32;
            }
        }
        then {
            routing-instance vpn-A;
        }
    }
    term vpnB {
        from {
            source-address {
                192.168.1.2/32;
```

```
            }
            destination-address {
                192.168.3.0/24;
            }
        }
        then routing-instance vpn-B;
    }
}
term internet {
    then accept;
}
```

You then configure the routing instances for VPN A and VPN B. Notice that these statements include all the required statements to define a Layer 3 VPN except for the `interface` statement.

```
[edit]
routing-instances {
    vpn-A {
        instance-type vrf;
        route-distinguisher 172.21.10.63:100;
        vrf-import vpn-A-import;
        vrf-export vpn-A-export;
    }
    vpn-B {
        instance-type vrf;
        route-distinguisher 172.21.10.63:200;
        vrf-import vpn-B-import;
        vrf-export vpn-B-export;
    }
}
```

Configuration on Router E

On Router E, configure a default route to reach the Internet. You should inject this route into the local IBGP mesh to provide an exit point from the network.

```
[edit]
routing-options {
    static {
        route 0.0.0.0/0 next-hop so-2/2/2.0 discard
```

```
    }
}
```

Configure the interface to Client E so that all incoming traffic on interface `fe-1/1/1.0` that matches the VPN policy is forwarded over VPN A:

```
[edit]
routing-instances {
    vpn-A {
        interface fe-1/1/1.0
        instance-type vrf;
        route-distinguisher 172.21.10.62:100;
        vrf-import vpn-A-import;
        vrf-export vpn-A-export;
        routing-options {
            static {
                route 192.168.2.0/24 next-hop fe-1/1/1.0;
            }
        }
    }
}
```

Configuration on Router F

Again, because the interfaces that use filter-based forwarding must not be bound to a VPN, you configure an alternate method to provide next-hop routes to the VRF table by defining an interface routing table group and sharing this group among all the routing tables. To provide a route back to the clients for normal inet.0 routing, you define a static route to include in inet.0 and redistribute the static route into BGP:

```
[edit]
routing-options {
    interface-routes {
        rib-group inet if-rib;
    }
    rib-groups {
        if-rib {
            import-rib [ inet.0 vpn-B.inet.0 ];
        }
```

```
    }
}
```

To direct traffic from VPN B to Client D, you configure the routing instance for VPN B on Router F. All incoming traffic from Client D on interface `so-3/3/3.0` is forwarded normally by means of the destination address based on the routes in inet.0.

```
[edit]
routing-instances {
    vpn-B {
        instance-type vrf;
        route-distinguisher 172.21.10.64:200;
        vrf-import vpn-B-import;
        vrf-export vpn-B-export;
        routing-options {
            static {
                route 192.168.3.0/24 next-hop so-3/3/3.0;
            }
        }
    }
}
```

IPv4 Traffic Over Layer 3 VPNs

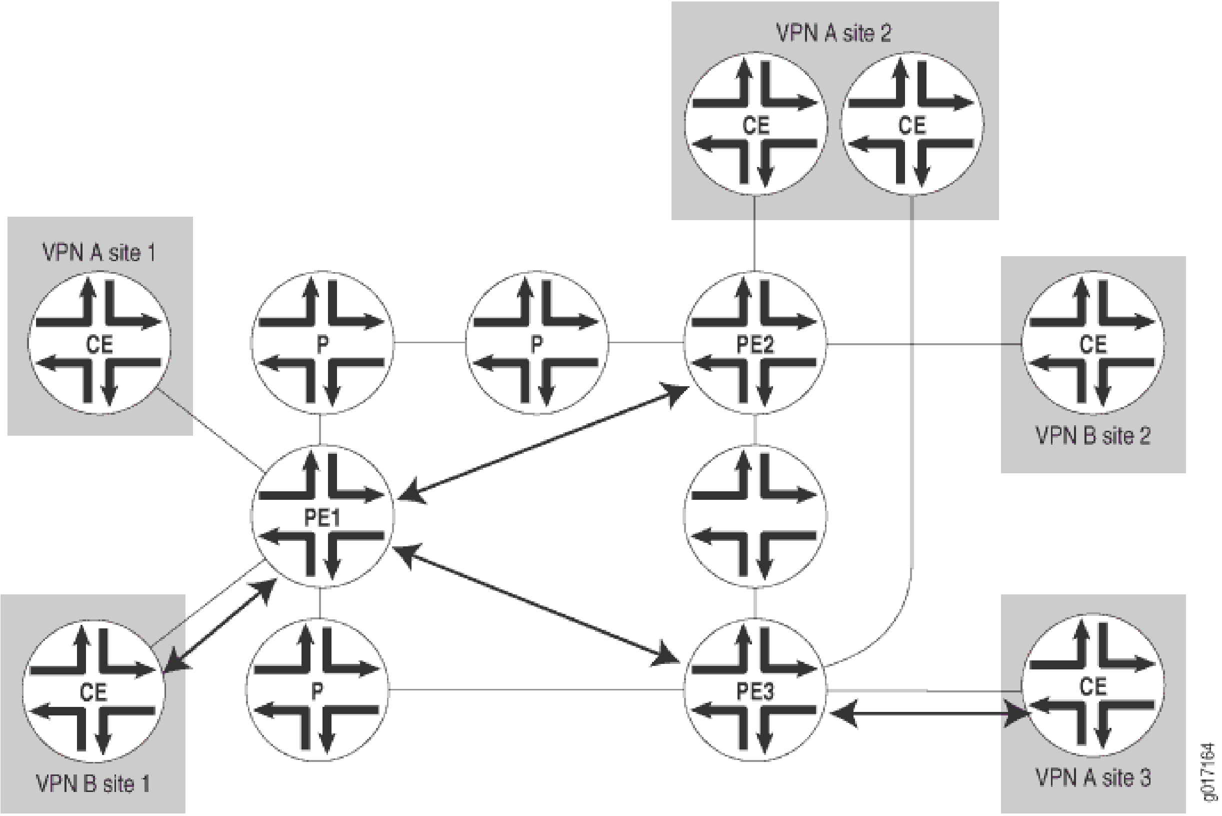
IN THIS SECTION

Understanding IPv4 Route Distribution in a Layer 3 VPN

IN THIS SECTION

Within a VPN, the distribution of VPN-IPv4 routes occurs between the PE and CE routers and between the PE routers (see Figure 20 on page 219).

Figure 20: Route Distribution Within a VPN



This section discusses the following topics:

Distribution of Routes from CE to PE Routers

A CE router announces its routes to the directly connected PE router. The announced routes are in IPv4 format. The PE router places the routes into the VRF table for the VPN. In the Junos OS, this is the *routing-instance-name*.inet.0 routing table, where `routing-instance-name` is the configured name of the VPN.

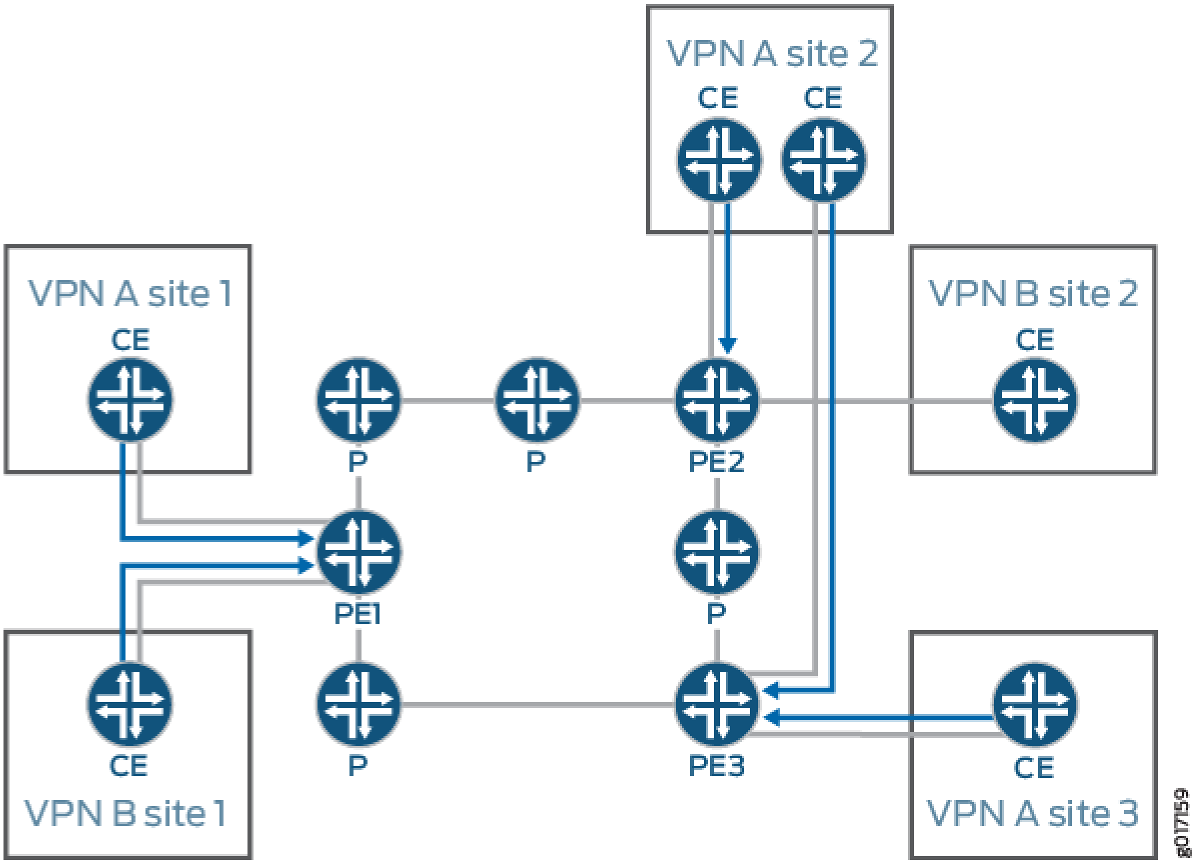
The connection between the CE and PE routers can be a remote connection (a WAN connection) or a direct connection (such as a Frame Relay or Ethernet connection).

CE routers can communicate with PE routers using one of the following:

- OSPF
- RIP
- BGP
- Static route

Figure 21 on page 221 illustrates how routes are distributed from CE routers to PE routers. Router PE1 is connected to two CE routers that are in different VPNs. Therefore, it creates two VRF tables, one for each VPN. The CE routers announce IPv4 routes. The PE router installs these routes into two different VRF tables, one for each VPN. Similarly, Router PE2 creates two VRF tables into which routes are installed from the two directly connected CE routers. Router PE3 creates one VRF table because it is directly connected to only one VPN.

Figure 21: Distribution of Routes from CE Routers to PE Routers



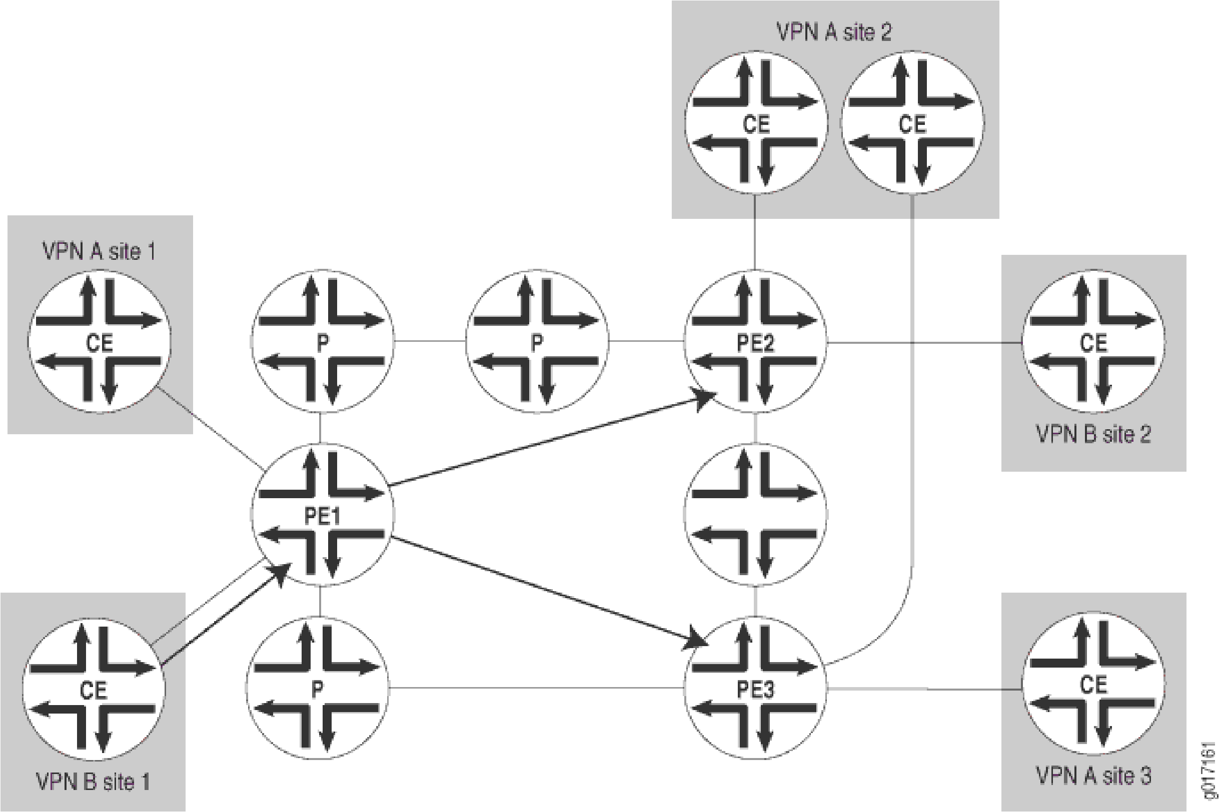
Distribution of Routes Between PE Routers

When one PE router receives routes advertised from a directly connected CE router, it checks the received route against the VRF export policy for that VPN. If it matches, the route is converted to VPN-IPv4 format—that is, the 8-byte route distinguisher is prepended to the 4-byte VPN prefix to form a 12-byte VPN-IPv4 address. The route is then tagged with a route target community. The PE router announces the route in VPN-IPv4 format to the remote PE routers for use by VRF import policies. The routes are distributed using IBGP sessions, which are configured in the provider's core network. If the route does not match, it is not exported to other PE routers, but can still be used locally for routing, for example, if two CE routers in the same VPN are directly connected to the same PE router.

The remote PE router places the route into its bgp.l3vpn.0 table if the route passes the import policy on the IBGP session between the PE routers. At the same time, it checks the route against the VRF import policy for the VPN. If it matches, the route distinguisher is removed from the route, and it is placed into the VRF table (the *routing-instance-name*.inet.0 table) in IPv4 format.

Figure 22 on page 222 illustrates how Router PE1 distributes routes to the other PE routers in the provider's core network. Router PE2 and Router PE3 each have VRF import policies that they use to determine whether to accept routes received over the IBGP sessions and install them in their VRF tables.

Figure 22: Distribution of Routes Between PE Routers



When a PE router receives routes advertised from a directly connected CE router (Router PE1 in Figure 22 on page 222), it uses the following procedure to examine the route, convert it to a VPN route, and distribute it to the remote PE routers:

1. The PE router checks the received route using the VRF export policy for that VPN.
2. If the received route matches the export policy, the route is processed as follows:
   a. The route is converted to VPN-IPv4 format—that is, the 8-byte route distinguisher is prepended to the 4-byte VPN prefix to form the 12-byte VPN-IPv4 address.
   b. A route target community is added to the route.
   c. The PE router advertises the route in VPN-IPv4 format to the remote PE routers. The routes are distributed using IBGP sessions, which are configured in the provider's core network.

3. If the route does not match the export policy, it is not exported to the remote PE routers, but can still be used locally for routing—for example,if two CE routers in the same VPN are directly connected to the same PE router.

When the remote PE router receives routes advertised from another PE router (Routers PE2 and PE3 in Figure 22 on page 222), it uses the following procedure to process the route:

1. If the route is accepted by the import policy on the IBGP session between the PE routers, the remote PE router places the route into its bgp.l3vpn.0 table.
2. The remote PE router checks the route's route target community against the VRF import policy for the VPN.
3. If it matches, the route distinguisher is removed from the route, and it is placed into the VRF table (the *routing-instance-name*.inet.0 table) in IPv4 format.

Distribution of Routes from PE to CE Routers

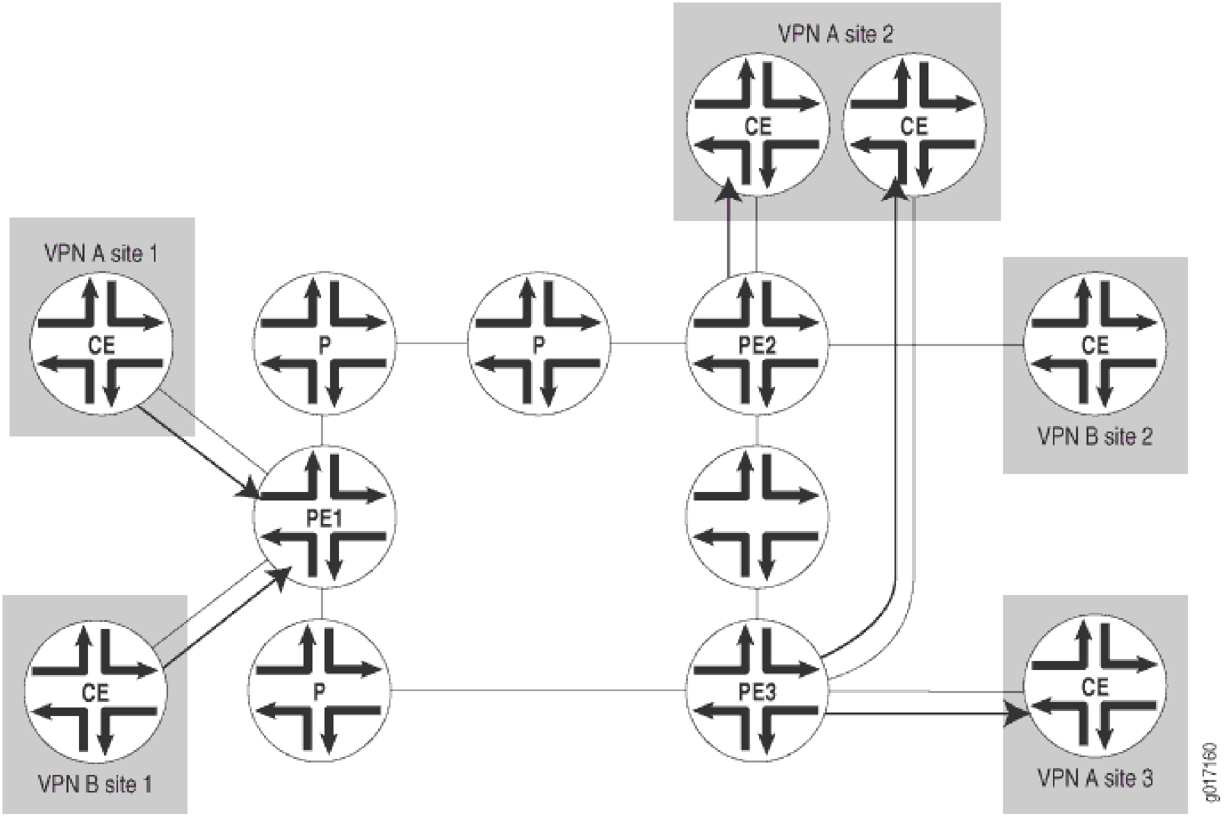
The remote PE router announces the routes in its VRF tables, which are in IPv4 format, to its directly connected CE routers.

PE routers can communicate with CE routers using one of the following routing protocols:

- OSPF
- RIP
- BGP
- Static route

Figure 23 on page 224 illustrates how the three PE routers announce their routes to their connected CE routers.

Figure 23: Distribution of Routes from PE Routers to CE Routers

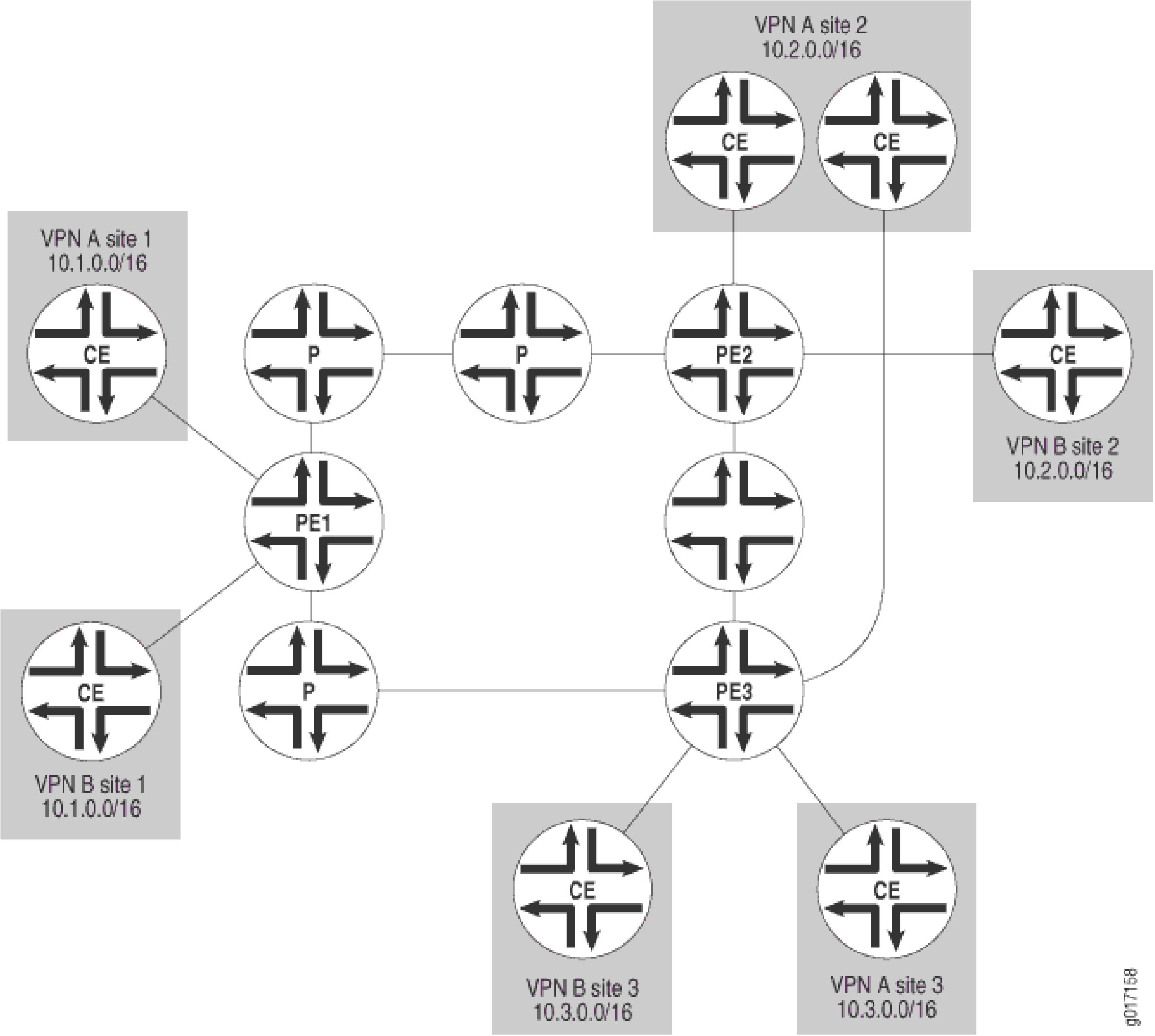


Understanding VPN-IPv4 Addresses and Route Distinguishers

Because Layer 3 VPNs connect private networks—which can use either public addresses or private addresses, as defined in RFC 1918 (*Address Allocation for Private Internets*)—over the public Internet infrastructure, when the private networks use private addresses, the addresses might overlap with the addresses of another private network.

Figure 24 on page 225 illustrates how private addresses of different private networks can overlap. Here, sites within VPN A and VPN B use the address spaces 10.1.0.0/16, 10.2.0.0/16, and 10.3.0.0/16 for their private networks.

Figure 24: Overlapping Addresses Among Different VPNs



To avoid overlapping private addresses, you can configure the network devices to use public addresses instead of private addresses. However, this is a large and complex undertaking. The solution provided in RFC 4364 uses the existing private network numbers to create a new address that is unambiguous. The new address is part of the VPN-IPv4 address family, which is a BGP address family added as an extension to the BGP protocol. In VPN-IPv4 addresses, a value that identifies the VPN, called a route distinguisher, is prefixed to the private IPv4 address, providing an address that uniquely identifies a private IPv4 address.

Only the PE routers need to support the VPN-IPv4 address extension to BGP. When an ingress PE router receives an IPv4 route from a device within a VPN, it converts it into a VPN-IPv4 route by adding the route distinguisher prefix to the route. The VPN-IPv4 addresses are used only for routes exchanged between PE routers. When an egress PE router receives a VPN-IPv4 route, it converts the VPN-IPv4 route back to an IPv4 route by removing the route distinguisher before announcing the route to its connected CE routers.

VPN-IPv4 addresses have the following format:

- Route distinguisher is a 6-byte value that you can specify in one of the following formats:
  - *`as-number`*`:`*`number`*, where *`as-number`* is an AS number (a 2-byte value) and *`number`* is any 4-byte value. The AS number can be in the range 1 through 65,535. We recommend that you use an Internet Assigned Numbers Authority (IANA)-assigned, nonprivate AS number, preferably the Internet service provider's (ISP's) own or the customer's own AS number.
  - *`ip-address`*`:`*`number`*, where *`ip-address`* is an IP address (a 4-byte value) and *`number`* is any 2-byte value. The IP address can be any globally unique unicast address. We recommend that you use the address that you configure in the `router-id` statement, which is a nonprivate address in your assigned prefix range.
- IPv4 address—4-byte address of a device within the VPN.

Figure 24 on page 225 illustrates how the AS number can be used in the route distinguisher. Suppose that VPN A is in AS 65535 and that VPN B is in AS 666 (both these AS numbers belong to the ISP), and suppose that the route distinguisher for Site 2 in VPN A is 65535:02 and that the route distinguisher for Site 2 in VPN B is 666:02. When Router PE2 receives a route from the CE router in VPN A, it converts it from its IP address of 10.2.0.0 to a VPN-IPv4 address of 65535:02:10.2.0.0. When the PE router receives a route from VPN B, which uses the same address space as VPN A, it converts it to a VPN-IPv4 address of 666:02:10.2.0.0.

If the IP address is used in the route distinguisher, suppose Router PE2's IP address is 172.168.0.1. When the PE router receives a route from VPN A, it converts it to a VPN-IPv4 address of 172.168.0.1:0:10.2.0.0/16, and it converts a route from VPN B to 172.168.0.0:1:10.2.0.0/16.

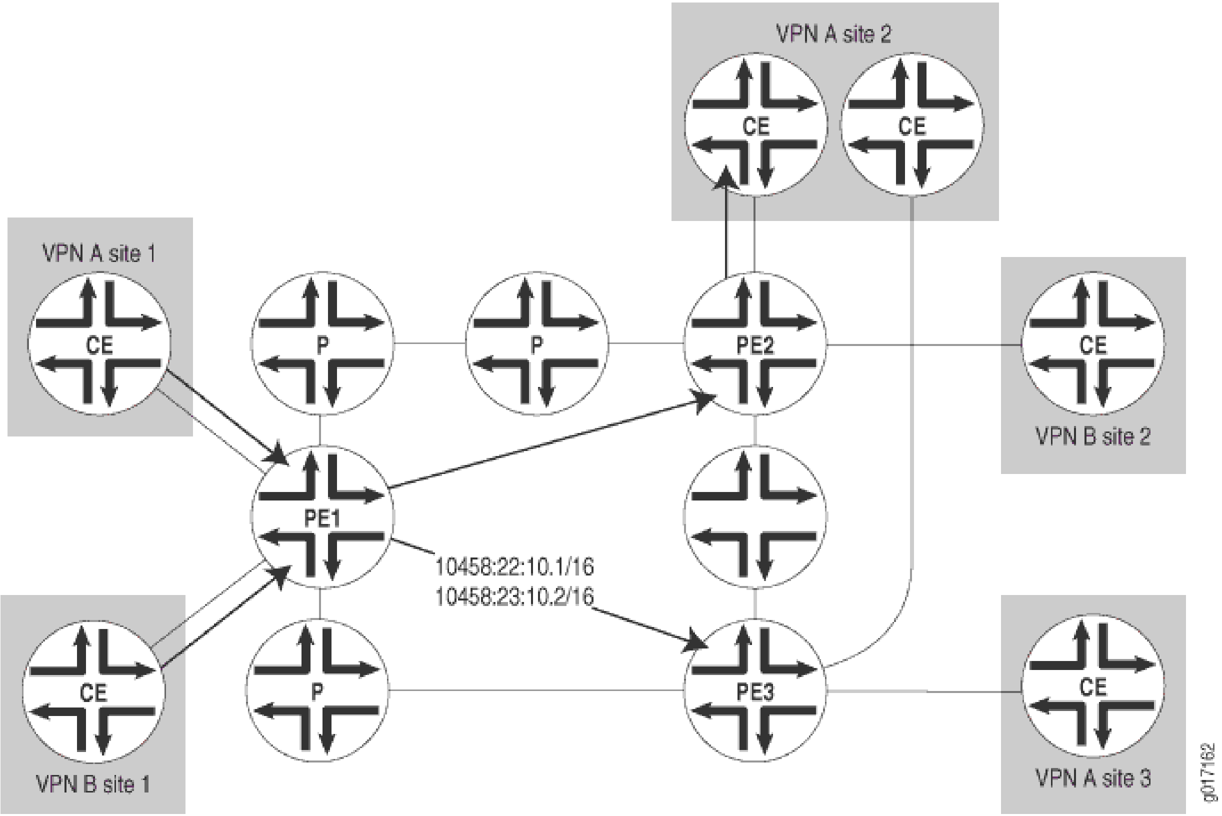
Route distinguishers are used only among PE routers to IPv4 addresses from different VPNs. The ingress PE router creates a route distinguisher and converts IPv4 routes received from CE routers into VPN-IPv4 addresses. The egress PE routers convert VPN-IPv4 routes into IPv4 routes before announcing them to the CE router.

Because VPN-IPv4 addresses are a type of BGP address, you must configure IBGP sessions between pairs of PE routers so that the PE routers can distribute VPN-IPv4 routes within the provider's core network. (All PE routers are assumed to be within the same AS.)

You define BGP communities to constrain the distribution of routes among the PE routers. Defining BGP communities does not, by itself, distinguish IPv4 addresses.

Figure 25 on page 227 illustrates how Router PE1 adds the route distinguisher 10458:22:10.1/16 to routes received from the CE router at Site 1 in VPN A and forwards these routes to the other two PE routers. Similarly, Router PE1 adds the route distinguisher 10458:23:10.2/16 to routes received by the CE router at Site 1 in VPN B and forwards these routes to the other PE routers.

Figure 25: Route Distinguishers



Configuring IPv4 Packet Forwarding for Layer 3 VPNs

You can configure the router to support packet forwarding for IPv4 traffic in Layer 2 and Layer 3 VPNs. Packet forwarding is handled in one of the following ways, depending on the type of helper service configured:

- BOOTP service—Clients send Bootstrap Protocol (BOOTP) requests through the router configured with BOOTP service to a server in the specified routing instance. The server recognizes the client address and sends a response back to the router configured with BOOTP service. This router forwards the reply to the correct client address in the specified routing instance.

- Other services—Clients send requests through the router configured with the service to a server in the specified routing instance. The server recognizes the client address and sends a response to the correct client address in the specified routing instance.

To enable packet forwarding for VPNs, include the `helpers` statement:

```
helpers {
    service {
        description description-of-service;
        server {
            address address {
                routing-instance routing-instance-names;
            }
        }
        interface interface-name {
            description description-of-interface;
            no-listen;
            server {
                address address {
                    routing-instance routing-instance-names;
                }
            }
        }
    }
}
```

You can include this statement at the following hierarchy levels:

- `[edit forwarding-options]`
- `[edit logical-systems logical-system-name forwarding-options]`
- `[edit routing-instances routing-instance-name forwarding-options]`

NOTE: You can enable packet forwarding for multiple VPNs. However, the client and server must be within the same VPN. Any Juniper Networks routing platforms with packet forwarding enabled along the path between the client and server must also reside within the same VPN.

The address and routing instance together constitute a unique server. This has implications for routers configured with BOOTP service, which can accept multiple servers.

For example, a BOOTP service can be configured as follows:

```
[edit forwarding-options helpers bootp]
server address 10.2.3.4 routing-instance [instance-A instance-B];
```

Even though the addresses are identical, the routing instances are different. A packet coming in for BOOTP service on `instance-A` is forwarded to `10.2.3.4` in the `instance-A` routing instance, while a packet coming in on `instance-B` is forwarded in the `instance-B` routing instance. Other services can only accept a single server, so this configuration does not apply in those cases.

Example: Configure a Basic MPLS-Based Layer 3 VPN

IN THIS SECTION

This example shows how to configure and validate a basic MPLS-based Layer 3 VPN on routers or switches running Junos OS. The IPv4 based example uses EBGP as the routing protocol between the provider and customer edge devices.

NOTE: Our content testing team has validated and updated this example.

You can deploy an MPLS-based Layer 3 virtual private network (VPN) using routers and switches running Junos OS to interconnect customer sites with Layer 3 connectivity. While static routing is supported, Layer 3 VPNs typically have the customer devices exchange routing information with the provider network and require support for IP protocols, i.e., IPv4 and/or IPv6.

This is in contrast with a Layer 2 VPN, where the customer devices may not be based on IP protocols, and where routing, if any, occurs between the customer edge (CE) devices. Unlike a Layer 3 VPN where the CE device interacts (peers) with the provider edge device, in a Layer 2 VPN the customer traffic passes transparently through the provider core with any routing protocols running end-to-end between the CE devices.

MPLS-based VPNs require baseline MPLS functionality in the provider network. Once basic MPLS is operational, you are able to configure VPNs that use label-switched paths (LSPs) for transport over the provider core.

The addition of VPN services does not affect the basic MPLS switching operations in the provider network. In fact, the provider (P) devices require only a baseline MPLS configuration because they are not VPN aware. VPN state is maintained only on the provider edge (PE) devices. This is a key reason why MPLS-based VPNs scale so well.

Requirements

This example uses the following software and hardware components:

- Junos OS Release 12.3 or later for routing and switching devices
  - Revalidated on Junos OS release 20.3R1
- Two Provider edge (PE) devices
- One provider (P) device
- Two customer edge (CE) devices

The example focuses on how to add a Layer 3 VPN to a pre-existing MPLS baseline. A basic MPLS configuration is provided in case your network does not already have MPLS deployed.

To support MPLS-based VPNs the underlying MPLS baseline must provide the following functionality:

- Core-facing and loopback interfaces operational with MPLS family support
- An interior gateway protocol such as OSPF or IS-IS to provide reachability between the loopback addresses of the provider (P and PE) devices
- An MPLS signalling protocol such as LDP or RSVP to signal LSPs
- LSPs established between PE device loopback addresses

LSPs are needed between each pair of PE devices that participate in a given VPN. Its a good idea to build LSPs between all PE devices to accommodate future VPN growth. You configure LSPs at the `[edit protocols mpls]` hierarchy level. Unlike an MPLS configuration for a circuit cross-connect (CCC) connection, you do not need to manually associate the LSP with the PE device’s customer-facing (edge) interface. Instead, Layer 3 VPNs use BGP signaling to advertise site reachability. This BGP signaling automates the mapping of remote VPN sites to LSP forwarding next hops. This means that with a Layer 3 VPN explicit mapping of an LSP to a PE device’s edge-facing interface is not required.

Overview and Topology

Layer 3 VPNs allow customers to leverage the service provider's technical expertise to ensure efficient site-to-site routing. The customer edge (CE) device typically uses a routing protocol, such as BGP or OSPF, to exchange routes with the service provider edge (PE) device. Static routing is supported for Layer 3 VPNs, but a dynamic routing protocol is generally preferred.

Definition of a VPN involves changes to the local and remote PE devices only. No additional configuration is needed on the provider devices (assuming they already have a working MPLS baseline), because these devices only provide basic MPLS switching functions. The CE devices do not use MPLS and require only a basic interface and routing protocol configuration so they can interact with the PE devices.

In a Layer 3 VPN you configure the CE devices to peer with the local PE device. This is in contrast to a Layer 2 VPN, where the CE devices peer to each other as if they were on a shared link, despite their being connected through an MPLS-based provider core.

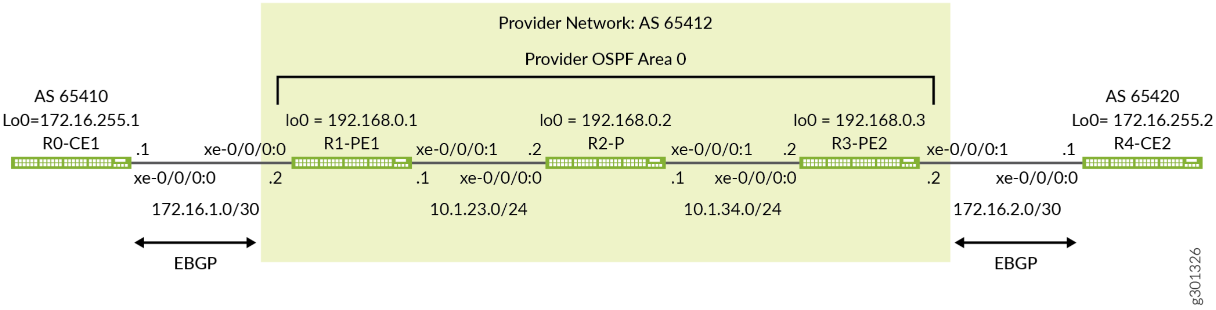
Once an MPLS baseline is in place, you must configure the following functionality on the PE devices to establish your MPLS-based Layer 3 VPN:

- A BGP group with `family inet-vpn unicast` support
- A routing instance with instance type `vrf` and a routing protocol definition that is compatible with the attached CE device
- The customer-facing interfaces on the PE devices configured with `family inet` along with an IPv4 address that places the interface on the same subnet as the attached CE device. If desired VLAN encapsulation and a corresponding VLAN ID can also be configured.

For proper end-to-end connectivity the CE device needs to be configured with a compatible IP subnet and routing protocol parameters to support peering with the PE device.

Figure 26 on page 232 shows the topology used in this example. The figure details the interface names, IP addressing, and routing protocols used in the provider and customer networks. It also highlights the peering relationship between the CE and PE devices. In this example you expect each CE device to form an EBGP peering session *to* the local PE device. Note that the provider network and both customer sites have an assigned autonomous system number to support BGP operation. In this example routing policy is applied at the CE devices to have them advertise the direct routes for their provider facing and loopback interfaces.

Figure 26: An MPLS-Based Layer 3 VPN with EBGP as the PE-CE Routing Protocol



Quick Configurations

IN THIS SECTION

Use the configurations in this section to quickly get your MPLS-based Layer 3 VPN up and running. The configurations include a functional MPLS baseline to support your Layer 3 VPN. This example focuses on the VPN aspects of the configuration. Refer to the following links for additional information on the baseline MPLS functionality used in this example:

- Configuring MPLS on Provider Edge EX8200 and EX4500 Switches Using Circuit Cross-Connect
- Configuring MPLS on EX8200 and EX4500 Provider Switches

CLI Quick Configuration

NOTE: The device configurations omit the management interface, static routes, system logging, system services, and user login information. These parts of the configuration vary by location and are not directly related to MPLS or VPN functionality.

Edit the following commands as needed for the specifics of your environment and paste them into the local CE (CE1) device terminal window when in configuration mode at the `[edit]` hierarchy:

The complete configuration for the CE1 device.

```
set system host-name ce1
set interfaces xe-0/0/0:0 description "Link from CE1 to PE1 for L3vpn"
set interfaces xe-0/0/0:0 unit 0 family inet address 172.16.1.1/30
set interfaces lo0 unit 0 family inet address 172.16.255.1/32
set routing-options router-id 172.16.255.1
set routing-options autonomous-system 65410
set protocols bgp group PE1 type external
set protocols bgp group PE1 export adv_direct
set protocols bgp group PE1 peer-as 65412
set protocols bgp group PE1 neighbor 172.16.1.2
set policy-options policy-statement adv_direct term 1 from protocol direct
set policy-options policy-statement adv_direct term 1 from route-filter 172.16.0.0/16 orlonger
set policy-options policy-statement adv_direct term 1 then accept
```

The complete configuration for PE1 device.

```
set system host-name pe1
set interfaces xe-0/0/0:0 description "Link from PE1 to CE1 for L3vpn"
set interfaces xe-0/0/0:0 unit 0 family inet address 172.16.1.2/30
set interfaces xe-0/0/0:1 description "Link from PE1 to p-router"
set interfaces xe-0/0/0:1 mtu 4000
set interfaces xe-0/0/0:1 unit 0 family inet address 10.1.23.1/24
set interfaces xe-0/0/0:1 unit 0 family mpls
set interfaces lo0 unit 0 family inet address 192.168.0.1/32
set routing-instances CE1_L3vpn protocols bgp group CE1 type external
set routing-instances CE1_L3vpn protocols bgp group CE1 peer-as 65410
set routing-instances CE1_L3vpn protocols bgp group CE1 neighbor 172.16.1.1
set routing-instances CE1_L3vpn instance-type vrf
set routing-instances CE1_L3vpn interface xe-0/0/0:0.0
set routing-instances CE1_L3vpn route-distinguisher 192.168.0.1:12
set routing-instances CE1_L3vpn vrf-target target:65412:12
set routing-options router-id 192.168.0.1
set routing-options autonomous-system 65412
set protocols bgp group ibgp type internal
set protocols bgp group ibgp local-address 192.168.0.1
set protocols bgp group ibgp family inet-vpn unicast
set protocols bgp group ibgp neighbor 192.168.0.3
set protocols mpls label-switched-path lsp_to_pe2 to 192.168.0.3
set protocols mpls interface xe-0/0/0:1.0
```

```
set protocols ospf traffic-engineering
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set protocols ospf area 0.0.0.0 interface xe-0/0/0:1.0
set protocols rsvp interface lo0.0
set protocols rsvp interface xe-0/0/0:1.0
```

The complete configuration for the P device.

```
set system host-name p
set interfaces xe-0/0/0:0 description "Link from p-router to PE1"
set interfaces xe-0/0/0:0 mtu 4000
set interfaces xe-0/0/0:0 unit 0 family inet address 10.1.23.2/24
set interfaces xe-0/0/0:0 unit 0 family mpls
set interfaces xe-0/0/0:1 description "Link from p-router to PE2"
set interfaces xe-0/0/0:1 mtu 4000
set interfaces xe-0/0/0:1 unit 0 family inet address 10.1.34.1/24
set interfaces xe-0/0/0:1 unit 0 family mpls
set interfaces lo0 unit 0 family inet address 192.168.0.2/32
set protocols mpls interface xe-0/0/0:0.0
set protocols mpls interface xe-0/0/0:1.0
set protocols ospf traffic-engineering
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set protocols ospf area 0.0.0.0 interface xe-0/0/0:0.0
set protocols ospf area 0.0.0.0 interface xe-0/0/0:1.0
set protocols rsvp interface lo0.0
set protocols rsvp interface xe-0/0/0:0.0
set protocols rsvp interface xe-0/0/0:1.0
```

The complete configuration for the PE2 device.

```
set system host-name pe2
set interfaces xe-0/0/0:1 description "Link from PE2 to CE2 for L3vpn"
set interfaces xe-0/0/0:1 unit 0 family inet address 172.16.2.2/30
set interfaces xe-0/0/0:0 description "Link from PE2 to p-router"
set interfaces xe-0/0/0:0 mtu 4000
set interfaces xe-0/0/0:0 unit 0 family inet address 10.1.34.2/24
set interfaces xe-0/0/0:0 unit 0 family mpls
set interfaces lo0 unit 0 family inet address 192.168.0.3/32
set routing-instances CE2_L3vpn protocols bgp group CE2 type external
set routing-instances CE2_L3vpn protocols bgp group CE2 peer-as 65420
set routing-instances CE2_L3vpn protocols bgp group CE2 neighbor 172.16.2.1
```

```
set routing-instances CE2_L3vpn instance-type vrf
set routing-instances CE2_L3vpn interface xe-0/0/0:1.0
set routing-instances CE2_L3vpn route-distinguisher 192.168.0.3:12
set routing-instances CE2_L3vpn vrf-target target:65412:12
set routing-options router-id 192.168.0.3
set routing-options autonomous-system 65412
set protocols bgp group ibgp type internal
set protocols bgp group ibgp local-address 192.168.0.3
set protocols bgp group ibgp family inet-vpn unicast
set protocols bgp group ibgp neighbor 192.168.0.1
set protocols mpls label-switched-path lsp_to_pe1 to 192.168.0.1
set protocols mpls interface xe-0/0/0:0.0
set protocols ospf traffic-engineering
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set protocols ospf area 0.0.0.0 interface xe-0/0/0:0.0
set protocols rsvp interface lo0.0
set protocols rsvp interface xe-0/0/0:0.0
```

The complete configuration for the CE2 device.

```
set system host-name ce2
set interfaces xe-0/0/0:0 description "Link from CE2 to PE2 for L3vpn"
set interfaces xe-0/0/0:0 unit 0 family inet address 172.16.2.1/30
set interfaces lo0 unit 0 family inet address 172.16.255.2/32
set routing-options router-id 172.16.255.2
set routing-options autonomous-system 65420
set protocols bgp group PE2 type external
set protocols bgp group PE2 export adv_direct
set protocols bgp group PE2 peer-as 65412
set protocols bgp group PE2 neighbor 172.16.2.2
set policy-options policy-statement adv_direct term 1 from protocol direct
set policy-options policy-statement adv_direct term 1 from route-filter 172.16.0.0/16 orlonger
set policy-options policy-statement adv_direct term 1 then accept
```

Be sure to commit the configuration changes on all devices when satisfied with your work. Congratulations on your new MPLS-based Layer 3 VPN! Refer to the "Verification" on page 249 section for the steps needed to confirm your Layer 3 VPN is working as expected.

Configure the Local PE (PE1) Device for a MPLS-Based Layer 3 VPN

IN THIS SECTION

This section covers the steps needed to configure the PE1 device for this example. The focus in on the PE devices because that is where the VPN configuration is housed. Refer to the "Quick Configurations" on page 232 section for the CE device and P device configurations used in this example.

Configure the MPLS Baseline (if needed)

Before you configure a Layer 3 VPN make sure the PE device has a working MPLS baseline. If you already having an MPLS baseline you can skip to the step-by-step procedure to add the Layer 3 VPN to the PE devices.

- Configure the hostname.

```
[edit]
user@pe1# set system host-name pe1
```

- Configure the core and loopback interfaces:

```
[edit]
user@pe1# set interfaces xe-0/0/0:1 description "Link from PE1 to P-router"
[edit]
user@pe1# set interfaces xe-0/0/0:1 mtu 4000
[edit]
user@pe1# set interfaces xe-0/0/0:1 unit 0 family inet address 10.1.23.1/24
[edit]
user@pe1# set interfaces xe-0/0/0:1 unit 0 family mpls
[edit]
user@pe1# set interfaces lo0 unit 0 family inet address 192.168.0.1/32
```

BEST PRACTICE: While a Layer 3 VPN can perform fragmentation at the ingress PE, its best practice to design the network so the CE can send a maximum sized frame without needing fragmentation. To ensure fragmentation does not occur the provider network should support the largest frame that the CE devices can generate *after* the MPLS and virtual routing and forwarding (VRF) labels are added by the PE device. This example leaves the CE devices at the default 1500-byte maximum transmission unit (MTU) while configuring the provider core to support a 4000 byte MTU. This ensures the CE devices cannot exceed the MTU in the provider's network even with the MPLS and VRF encapsulation overhead.

- Configure the protocols:

NOTE: Traffic engineering is supported for RSVP-signaled LSPs but is not required for basic MPLS switching or VPN deployment. The provided MPLS baseline uses RSVP to signal LSPs, and enables traffic engineering for OSPF. However, no path constraints are configured so you expect the LSPs to be routed over the interior gateway protocol's shortest path.

```
[edit]
user@pe1# set protocols ospf area 0.0.0.0 interface lo0.0 passive
[edit]
user@pe1# set protocols ospf area 0.0.0.0 interface xe-0/0/0:1.0
[edit]
user@pe1# set protocols ospf traffic-engineering
[edit]
user@pe1# set protocols mpls interface xe-0/0/0:1.0
[edit]
user@pe1# set protocols rsvp interface lo0.0
[edit]
user@pe1# set protocols rsvp interface xe-0/0/0:1.0
```

- Define the LSP to the remote PE device's loopback address:

```
[edit]
user@pe1#  set protocols mpls label-switched-path lsp_to_pe2 to 192.168.0.3
```

The MPLS baseline is now configured on the PE1 device. Keep going to configure the Layer 3 VPN.

Procedure

Step-by-Step Procedure

Follow these steps to configure the PE1 device for a Layer 3 VPN.

1. Configure the customer-facing interface:

   TIP: You can configure both an MPLS-based Layer 2 VPN and an MPLS-based Layer 3 VPN on the same PE device. However, you cannot configure the same customer edge-facing interface to support both a Layer 2 VPN and a Layer 3 VPN.

```
[edit interfaces]
user@pe1# set xe-0/0/0:0 description "Link from PE1 to CE1 for L3vpn"
[edit]
user@pe1# set xe-0/0/0:0 unit 0 family inet address 172.16.1.2/30
```

2. Configure a BGP group for the peering between the local and remote PE devices. Use the PE device's loopback address as the local address, and enable the `inet-vpn unicast` address family to support Layer 3 VPN route exchange. A routing policy for BGP is not needed on the PE devices in this example. By default the PE devices readvertise into IBGP the routes they learn over their EBGP peering to the CE device.

```
[edit protocols bgp]
user@pe1# set group ibgp local-address 192.168.0.1
[edit protocols bgp]
user@pe1# set group ibgp family inet-vpn unicast
```

   TIP: You can specify other address families if the PE to PE IBGP session needs to support non-VPN route exchange, such as regular IPv4 or IPv6 routes using the `inet` or `inet6` families, respectively.

3. Configure the BGP group type as internal.

```
[edit protocols bgp]
user@pe1# set group ibgp type internal
```

4. Configure the remote PE device’s loopback address as a BGP neighbor.

```
[edit protocols bgp]
user@pe1# set group ibgp neighbor 192.168.0.3
```

5. Configure the router ID to match its loopback address, and define the BGP autonomous system number needed for BGP peering.

```
[edit routing-options]
user@pe1# set router-id 192.168.0.1
[edit routing-options]
user@pe1# set autonomous-system 65412
```

6. Configure the routing instance. Specify an instance name of *CE1_L3vpn*, and configure an `instance-type` of `vrf`.

```
[edit routing-instances]
user@pe1# set CE1_L3vpn instance-type vrf
```

7. Configure the PE device’s customer-facing interface to belong to the routing instance.

```
[edit routing-instances]
user@pe1# set CE1_L3vpn interface xe-0/0/0:0.0
```

8. Configure the routing instance’s route distinguisher. This setting is used to distinguish the routes sent from a particular VRF on a particular PE device. It should be unique for each routing instance on each PE device.

```
[edit routing-instances]
user@pe1# set CE1_L3vpn route-distinguisher 192.168.0.1:12
```

9. Configure the instance’s virtual routing and forwarding (VRF) table route target. The `vrf-target` statement adds the specified community tag to all advertised routes while automatically matching

the same value for route import. Configuring matching route targets on the PE devices that share a given VPN is required for proper route exchange.

```
[edit routing-instances]
user@pe1# set CE1_L3vpn vrf-target target:65142:12
```

NOTE: You can create more complex policies by explicitly configuring VRF import and export policies using the import and export options. See *vrf-import* and *vrf-export* for details.

10. Configure the routing instance to support EBGP peering to the CE1 device. Direct interface peering to the CE1 end of the VRF link is used, and CE1's autonomous system number is correctly specified with the `peer-as` parameter.

```
[edit routing-instances]
user@pe1# set CE1_L3vpn protocols bgp group CE1 type external
[edit routing-instances]
user@pe1# set CE1_L3vpn protocols bgp group CE1 peer-as 65410
[edit routing-instances]
user@pe1# set CE1_L3vpn protocols bgp group CE1 neighbor 172.16.1.1
```

11. Commit your changes at the PE1 device and return to CLI operational mode.

```
[edit]
user@pe1# commit and-quit
```

Results

Display the results of the configuration on the PE1 device. The output reflects only the functional configuration added in this example.

```
user@pe1> show configuration
```

```
interfaces {
    xe-0/0/0:0 {
        description "Link from PE1 to CE1 for L3vpn";
```

```
        unit 0 {
             family inet {
        }
    }
    unit 0 {
        family inet {
            address 10.1.23.1/24;
        }
        family mpls;
    }
    ge-0/0/1 {
        description "Link from PE1 to P-router";
        mtu 4000;
        unit 0 {
            family inet {
                address 10.1.23.1/24;
            }
            family mpls;
        }
    }
    lo0 {
        unit 0 {
            family inet {
                address 192.168.0.1/32;
            }
        }
    }
}
routing-instances {
    CE1_L3vpn {
        protocols {
            bgp {
                group CE1 {
                    type external;
                    peer-as 65410;
                    neighbor 172.16.1.1;
                }
            }
        }
        instance-type vrf;
        interface xe-0/0/0:0.0;
        route-distinguisher 192.168.0.1:12;
        vrf-target target:65412:12;
```

```
    }
}
routing-options {
    router-id 192.168.0.1;
    autonomous-system 65412;
}
protocols {
    bgp {
        group ibgp {
            type internal;
            local-address 192.168.0.1;
            family inet-vpn {
                unicast;
            }
            neighbor 192.168.0.3;
        }
    }
    mpls {
        label-switched-path lsp_to_pe2 {
            to 192.168.0.3;
        }
        interface xe-0/0/0:1.0;
    }
    ospf {
        traffic-engineering;
        area 0.0.0.0 {
            interface lo0.0 {
                passive;
            }
            interface xe-0/0/0:1.0;
        }
    }
    rsvp {
        interface lo0.0;
        interface xe-0/0/0:1.0;
    }
}
```

Configure the Remote PE (PE2) Device for a MPLS-Based Layer 3 VPN

IN THIS SECTION

This section covers the steps needed to configure the PE1 device for this example. The focus in on the PE devices because that is where the VPN configuration is housed. Refer to the "Quick Configurations" on page 232 section for the CE device and P device configurations used in this example.

Configure the MPLS Baseline (if needed)

Before you configure a Layer 3 VPN make sure the PE device has a working MPLS baseline. If you already having an MPLS baseline you can skip to the step-by-step procedure to add the Layer 3 VPN to the PE devices.

- Configure the hostname.

```
[edit]
user@pe2# set system host-name pe2
```

- Configure the core and loopback interfaces:

```
[edit]
user@pe2# set interfaces xe-0/0/0:0 description "Link from PE2 to P-router"
[edit]
user@pe2# set interfaces xe-0/0/0:0 mtu 4000
[edit]
user@pe2# set interfaces xe-0/0/0:0 unit 0 family inet address 10.1.34.2/24
[edit]
user@pe2# set interfaces xe-0/0/0:0 unit 0 family mpls
[edit]
user@pe2# set interfaces lo0 unit 0 family inet address 192.168.0.3/32
```

BEST PRACTICE: While a Layer 3 VPN can perform fragmentation at the ingress PE, its best practice to design the network so the CE can send a maximum sized frame without needing fragmentation. To ensure fragmentation does not occur the provider network should support the largest frame that the CE devices can generate *after* the MPLS and virtual routing and forwarding (VRF) labels are added by the PE device. This example leaves the CE devices at the default 1500-byte maximum transmission unit (MTU) while configuring the provider core to support a 4000 byte MTU. This ensures the CE devices cannot exceed the MTU in the provider's network even with the MPLS and VRF encapsulation overhead.

- Configure the protocols:

NOTE: Traffic engineering is supported for RSVP-signaled LSPs but is not required for basic MPLS switching or VPN deployment. The provided MPLS baseline uses RSVP to signal LSPs, and enables traffic engineering for OSPF. However, no path constraints are configured so you expect the LSPs to be routed over the interior gateway protocol's shortest path.

```
[edit]
user@pe2# set protocols ospf area 0.0.0.0 interface lo0.0 passive
[edit]
user@pe2# set protocols ospf area 0.0.0.0 interface xe-0/0/0:0.0
[edit]
user@pe2# set protocols ospf traffic-engineering
[edit]
user@pe2# set protocols mpls interface xe-0/0/0:0.0
[edit]
user@pe2# set protocols rsvp interface lo0.0
[edit]
user@pe2# set protocols rsvp interface xe-0/0/0:0.0
```

- Define the LSP to the remote PE device's loopback address:

```
[edit]
user@pe2# set protocols mpls label-switched-path lsp_to_pe1 to 192.168.0.1
```

The MPLS baseline is now configured on the PE1 device. Keep going to configure the Layer 3 VPN.

Procedure

Step-by-Step Procedure

Follow these steps to configure the PE2 device for a Layer 3 VPN.

1. Configure the customer-facing interface:

   > **TIP**: You can configure both an MPLS-based Layer 2 VPN and an MPLS-based Layer 3 VPN on the same PE device. However, you cannot configure the same customer edge-facing interface to support both a Layer 2 VPN and a Layer 3 VPN.

   ```
   [edit interfaces]
   user@pe2# set xe-0/0/0:1 description "Link from PE2 to CE2 for L3vpn"
   [edit]
   user@pe2# set xe-0/0/0:1 unit 0 family inet address 172.16.2.2/30
   ```

2. Configure a BGP group for the peering between the local and remote PE devices. Use the PE device's loopback address as the local address, and enable the `inet-vpn unicast` address family to support Layer 3 VPN route exchange.

   ```
   [edit protocols bgp]
   user@pe1# set group ibgp local-address 192.168.0.1
   [edit protocols bgp]
   user@pe1# set group ibgp family inet-vpn unicast
   ```

   > **TIP**: You can specify other address families if the PE to PE IBGP session needs to support non-VPN route exchange, such as regular IPv4 or IPv6 routes using the `inet` or `inet6` families, respectively.

3. Configure the BGP group type as internal.

   ```
   [edit protocols bgp]
   user@pe2# set group ibgp type internal
   ```

4. Configure the PE1 device's loopback address as a BGP neighbor.

```
[edit protocols bgp]
user@pe2# set group ibgp neighbor 192.168.0.1
```

5. Configure the router ID to match its loopback address, and define the BGP autonomous system number.

```
[edit routing-options]
user@pe2# set routing-options router-id 192.168.0.3
[edit routing-options]
user@pe2# set autonomous-system 65412
```

6. Configure the routing instance. Specify an instance name of *CE2_L3vpn*, with an `instance-type` of `vrf`.

```
[edit routing-instances]
user@pe2# set CE2_L3vpn instance-type vrf
```

7. Configure the PE device's customer-facing interface to belong to the routing instance.

```
[edit routing-instances]
user@pe2# set CE2_L3vpn interface xe-0/0/0:1.0
```

8. Configure the routing instance's route distinguisher. This setting is used to distinguish the routes sent from a particular VRF on a particular PE device. It should be unique for each routing instance on each PE device.

```
[edit routing-instances]
user@pe2# set CE2_L3vpn route-distinguisher 192.168.0.3:12
```

9. Configure the instance's virtual routing and forwarding (VRF) table route target. The `vrf-target` statement adds the specified community tag to all advertised routes while automatically matching

the same value for route import. Configuring matching route targets on the PE devices that share a given VPN is required for proper route exchange.

```
[edit routing-instances]
user@pe2# set CE2_L3vpn vrf-target target:65412:12
```

NOTE: You can create more complex policies by explicitly configuring VRF import and export policies using the import and export options. See *vrf-import* and *vrf-export* for details.

10. Configure the routing instance to support EBGP peering to the CE2 device. Direct interface peering to the CE2 end of the VRF link is used, and CE2's autonomous system number is correctly specified with the `peer-as` parameter.

```
[edit routing-instances]
user@pe2# set CE2_L3vpn protocols bgp group CE2 type external
[edit routing-instances]
user@pe2# set CE2_L3vpn protocols bgp group CE2 peer-as 65420
[edit routing-instances]
user@pe2# set CE2_L3vpn protocols bgp group CE2 neighbor 172.16.2.1
```

11. Commit your changes at the PE2 device and return to CLI operational mode.

```
[edit]
user@pe2# commit and-quit
```

Results

Display the results of the configuration on the PE2 device. The output reflects only the functional configuration added in this example.

```
user@pe2> show configuration
```

```
interfaces {
    xe-0/0/0:0 {
        description "Link from PE2 to p-router";
```

```
        mtu 4000;
        unit 0 {
            family inet {
                address 10.1.34.2/24;
            }
            family mpls;
        }
    }
    xe-0/0/0:1 {
        description "Link from PE2 to CE2 for L3vpn";
        unit 0 {
            family inet {
                address 172.16.2.2/30;
            }
        }
    }
    lo0 {
        unit 0 {
            family inet {
                address 192.168.0.3/32;
            }
        }
    }
}
routing-instances {
    CE2_L3vpn {
        protocols {
            bgp {
                group CE2 {
                    type external;
                    peer-as 65420;
                    neighbor 172.16.2.1;
                }
            }
        }
        instance-type vrf;
        interface xe-0/0/0:1.0;
        route-distinguisher 192.168.0.3:12;
        vrf-target target:65412:12;
    }
}
routing-options {
    router-id 192.168.0.3;
```

```
    autonomous-system 65412;
}
protocols {
    bgp {
        group ibgp {
            type internal;
            local-address 192.168.0.3;
            family inet-vpn {
                unicast;
            }
            neighbor 192.168.0.1;
        }
    }
    mpls {
        label-switched-path lsp_to_pe1 {
            to 192.168.0.1;
        }
        interface xe-0/0/0:0.0;
    }
    ospf {
        traffic-engineering;
        area 0.0.0.0 {
            interface lo0.0 {
                passive;
            }
            interface xe-0/0/0:0.0;
        }
    }
    rsvp {
        interface lo0.0;
        interface xe-0/0/0:0.0;
    }
}
```

Verification

IN THIS SECTION

Perform these tasks to verify that the MPLS-based Layer 3 VPN works properly:

Verify Provider OSPF Adjacencies and Route Exchange

Purpose

Confirm the OSPF protocol is working properly in the provider network by verifying adjacency status and OSPF learned routes to the loopback addresses of the remote provider devices. Proper IGP operation is critical for the successful establishment of MPLS LSPs.

Action

```
user@pe1> show ospf neighbor

Address          Interface              State          ID               Pri  Dead
10.1.23.2        xe-0/0/0:1.0           Full           192.168.0.2      128    37
```

```
user@pe1> show route protocol ospf | match 192.168

192.168.0.2/32     *[OSPF/10] 1w1d 23:59:43, metric 1
192.168.0.3/32     *[OSPF/10] 1w1d 23:59:38, metric 2
```

Meaning

The output shows that the PE1 device has established an OSPF adjacency to the P device (`192.168.0.2`). It also shows that the P and remote PE device loopback addresses (`192.168.0.2`) and (`192.168.0.3`) are correctly learned via OSPF at the local PE device.

Verify MPLS and RSVP Interface Settings

Purpose

Verify that the RSVP and MPLS protocols are configured to operate on the PE device’s core-facing interface. This step also verifies that `family mpls` is correctly configured at the unit level of the PE device’s core-facing interface.

Action

```
user@pe1> show mpls interface

Interface        State       Administrative groups (x: extended)
xe-0/0/0:1.0     Up         <none>
```

```
user@pe1> show rsvp interface

RSVP interface: 2 active
                          Active  Subscr- Static      Available   Reserved    Highwater
Interface          State  resv    iption  BW          BW          BW          mark
lo0.0                  Up       0   100%  0bps        0bps        0bps        0bps
xe-0/0/0:1.0           Up       1   100%  10Gbps      10Gbps      0bps        0bps
```

Meaning

The output shows that MPLS and RSVP are correctly configured on the local PE device’s core and loopback interfaces.

Verify RSVP Signaled LSPs

Purpose

Verify that RSVP signaled ingress and egress LSPs are correctly established between the PE device’s loopback addresses.

Action

```
user@pe1> show rsvp session

Ingress RSVP: 1 sessions
To              From            State   Rt Style Labelin Labelout LSPname
192.168.0.3     192.168.0.1     Up       0  1 FF       -       17 lsp_to_pe2
Total 1 displayed, Up 1, Down 0

Egress RSVP: 1 sessions
To              From            State   Rt Style Labelin Labelout LSPname
192.168.0.1     192.168.0.3     Up       0  1 FF       3        - lsp_to_pe1
Total 1 displayed, Up 1, Down 0

Transit RSVP: 0 sessions
Total 0 displayed, Up 0, Down 0
```

Meaning

The output shows that both the ingress and egress RSVP sessions are correctly established between the PE devices. Successful LSP establishment indicates the MPLS baseline is operational.

Verify BGP Session Status

Purpose

Verify that the IBGP session between the PE devices is correctly established with support for Layer 3 VPN network layer reachability information (NLRI). In this step you also confirm the local PE to CE EBGP session is established and correctly configured to exchange IPv4 routes.

Action

```
user@pe1> show bgp summary

Groups: 2 Peers: 2 Down peers: 0
Table          Tot Paths  Act Paths Suppressed    History Damp State    Pending
inet.0
                       0          0          0          0          0          0
bgp.l3vpn.0
                       2          2          0          0          0          0
```

```
Peer                     AS      InPkt     OutPkt    OutQ   Flaps Last Up/Dwn State|#Active/
Received/Accepted/Damped...
172.16.1.1            65410      26038      25938       0       0 1w1d 3:12:32 Establ
  CE1_L3vpn.inet.0: 1/2/2/0
192.168.0.3           65412         19         17       0       0        6:18 Establ
  CE1_L3vpn.inet.0: 2/2/2/0
  bgp.l3vpn.0: 2/2/2/0
```

Meaning

The output shows the IBGP session to the remote PE device (`192.168.0.3`) has been correctly established (`Establ`), and through the `Up/Dwn` field, how long the session has been in the current state (`6:18`). The `flaps` field confirms that no state transitions have occurred (`0`), indicating the session is stable. Also note that Layer 3 VPN routes (NLRI) have been learned from the remote PE as shown by the presence of a `bgp.l3vpn.0` table.

The display also confirms the EBGP session to the local CE1 device (`172.16.1.1`) is established and that IPv4 routes have been received from the CE1 device and installed in the CE1 device routing instance (`CE1_L3vpn.inet.0`)

This output confirms that the BGP peering between the PE devices, and to the CE device, is working properly to support your Layer 3 VPN.

Verify Layer 3 VPN Routes in the Routing Table

Purpose

Confirm that the routing table on the PE1 device is populated with Layer 3 VPN routes advertised by the remote PE. These routes are used to forward traffic to the remote CE device.

Action

```
user@pe1> show route table bgp.l3vpn.0

bgp.l3vpn.0: 2 destinations, 2 routes (2 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

192.168.0.3:12:172.16.2.0/30
                   *[BGP/170] 00:22:45, localpref 100, from 192.168.0.3
                      AS path: I, validation-state: unverified
                    >  to 10.1.23.2 via xe-0/0/0:1.0, label-switched-path lsp_to_pe2
```

```
192.168.0.3:12:172.16.255.2/32
                   *[BGP/170] 00:22:43, localpref 100, from 192.168.0.3
                      AS path: 65420 I, validation-state: unverified
                    >  to 10.1.23.2 via xe-0/0/0:1.0, label-switched-path lsp_to_pe2
```

```
user@pe1> show route table CE1_L3vpn.inet.0

CE1_L3vpn.inet.0: 5 destinations, 6 routes (5 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

172.16.1.0/30      *[Direct/0] 1w1d 03:29:44
                    >  via xe-0/0/0:0.0
                    [BGP/170] 1w1d 03:29:41, localpref 100
                      AS path: 65410 I, validation-state: unverified
                    >  to 172.16.1.1 via xe-0/0/0:0.0
172.16.1.2/32      *[Local/0] 1w1d 03:29:44
                       Local via xe-0/0/0:0.0
172.16.2.0/30      *[BGP/170] 00:23:28, localpref 100, from 192.168.0.3
                      AS path: I, validation-state: unverified
                    >  to 10.1.23.2 via xe-0/0/0:1.0, label-switched-path lsp_to_pe2
172.16.255.1/32    *[BGP/170] 1w1d 03:29:41, localpref 100
                      AS path: 65410 I, validation-state: unverified
                    >  to 172.16.1.1 via xe-0/0/0:0.0
172.16.255.2/32    *[BGP/170] 00:23:26, localpref 100, from 192.168.0.3
                      AS path: 65420 I, validation-state: unverified
                    >  to 10.1.23.2 via xe-0/0/0:1.0, label-switched-path lsp_to_pe2
```

Meaning

The `show route table bgp.l3vpn.0` command displays the Layer 3 VPN routes that have been received from the remote PE device. The `show route table CE1_L3vpn.inet.0` command lists the all routes that have been imported into the `CE1_L3vpn` routing instance. These entries represent the routes learned from the local EBGP peering to the CE1 device, in addition to those routes received from the remote PE2 device with a matching route target.

Both tables show the remote Layer 3 VPN routes are correctly associated with the `lsp_to_pe2` LSP as a forwarding next hop. The outputs confirm the local PE device has learned the routes associated with the remote CE2 location from the PE2 device. It also shows that the local PE will forward Layer 3 VPN traffic to the remote PE2 device using MPLS transport over the provider network.

Ping the Remote PE Device Using the Layer 3 VPN Connection

Purpose

Verify Layer 3 VPN connectivity between the local and remote PE devices using ping. This command verifies the Layer 3 VPN routing and MPLS forwarding operation between the PE devices.

Action

```
user@pe1> ping routing-instance CE1_L3vpn 172.16.2.2 source 172.16.1.2 count 2

PING 172.16.2.2 (172.16.2.2): 56 data bytes
64 bytes from 172.16.2.2: icmp_seq=0 ttl=61 time=128.235 ms
64 bytes from 172.16.2.2: icmp_seq=1 ttl=61 time=87.597 ms

--- 172.16.2.2 ping statistics ---
2 packets transmitted, 2 packets received, 0% packet loss
round-trip min/avg/max/stddev = 87.597/107.916/128.235/20.319 m
```

Meaning

The output confirms that the Layer 3 VPN control and forwarding planes are operating correctly between the PE devices.

Verify End-to-End Operation of the CE Devices Over the Layer 3 VPN

Purpose

Verify Layer 3 VPN connectivity between the CE devices. This step confirms the CE devices have operational interfaces and are properly configured for EBGP based Layer 3 connectivity. This is done by verifying the local CE1 device has learned the remote CE device's routes, and by confirming the CE devices are able to pass traffic end-to-end between their loopback addresses.

Action

```
user@ce1> show route protocol bgp

inet.0: 12 destinations, 12 routes (12 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both
```

```
172.16.2.0/30      *[BGP/170] 00:40:50, localpref 100
                      AS path: 65412 I, validation-state: unverified
                    >  to 172.16.1.2 via xe-0/0/0:0.0
172.16.255.2/32    *[BGP/170] 00:40:49, localpref 100
                      AS path: 65412 65420 I, validation-state: unverified
                    >  to 172.16.1.2 via xe-0/0/0:0.0

inet6.0: 1 destinations, 1 routes (1 active, 0 holddown, 0 hidden)
```

```
user@ce1> ping 172.16.255.2 source 172.16.255.1 size 1472 do-not-fragment count 2

PING 172.16.255.2 (172.16.255.2): 1472 data bytes
1480 bytes from 172.16.255.2: icmp_seq=0 ttl=61 time=79.245 ms
1480 bytes from 172.16.255.2: icmp_seq=1 ttl=61 time=89.125 ms

--- 172.16.255.2 ping statistics ---
2 packets transmitted, 2 packets received, 0% packet loss
round-trip min/avg/max/stddev = 79.245/84.185/89.125/4.940 ms
```

Meaning

The output shows that Layer 3 VPN based connectivity is working correctly between the CE devices. The local CE device has learned the remote CE device's VRF interface and loopback routes through BGP. The ping is generated to the loopback address of the remote CE device, and is sourced from the local CE device's loopback address using the `source 172.16.255.1` argument. Adding the `do-not-fragment` and `size 1472` switches confirms that the CE devices are able to pass 1500-byte IP packets without evoking fragmentation in the local PE device.

NOTE: The `size 1472` argument added to the `ping` command generates 1472 bytes of echo data. An additional 8 bytes of Internet Control Message Protocol (ICMP) and 20 bytes of IP header are added to bring the total payload size to 1500-bytes. Adding the `do-not-fragment` switch ensures that the local CE and PE devices cannot perform fragmentation. This ping method confirms that fragmentation is not needed when exchanging standard 1500-byte maximum length Ethernet frames between the CE devices.

These results confirm the MPLS-based Layer 3 VPN is working correctly.

SEE ALSO

Example: Configuring MPLS on EX8200 and EX4500 Switches

Example: Configure MPLS-Based Layer 2 VPNs

RELATED DOCUMENTATION

Routing Policies, Firewall Filters, and Traffic Policers User Guide

Example: Configure a Basic MPLS-Based Layer 3 VPN

IN THIS SECTION

This example shows how to configure and validate a basic MPLS-based Layer 3 VPN on routers or switches running Junos OS. The IPv4 based example uses EBGP as the routing protocol between the provider and customer edge devices.

NOTE: Our content testing team has validated and updated this example.

You can deploy an MPLS-based Layer 3 virtual private network (VPN) using routers and switches running Junos OS to interconnect customer sites with Layer 3 connectivity. While static routing is supported, Layer 3 VPNs typically have the customer devices exchange routing information with the provider network and require support for IP protocols, i.e., IPv4 and/or IPv6.

This is in contrast with a Layer 2 VPN, where the customer devices may not be based on IP protocols, and where routing, if any, occurs between the customer edge (CE) devices. Unlike a Layer 3 VPN where the CE device interacts (peers) with the provider edge device, in a Layer 2 VPN the customer traffic passes transparently through the provider core with any routing protocols running end-to-end between the CE devices.

MPLS-based VPNs require baseline MPLS functionality in the provider network. Once basic MPLS is operational, you are able to configure VPNs that use label-switched paths (LSPs) for transport over the provider core.

The addition of VPN services does not affect the basic MPLS switching operations in the provider network. In fact, the provider (P) devices require only a baseline MPLS configuration because they are not VPN aware. VPN state is maintained only on the provider edge (PE) devices. This is a key reason why MPLS-based VPNs scale so well.

Requirements

This example uses the following software and hardware components:

- Junos OS Release 12.3 or later for routing and switching devices
  - Revalidated on Junos OS release 20.3R1
- Two Provider edge (PE) devices
- One provider (P) device
- Two customer edge (CE) devices

The example focuses on how to add a Layer 3 VPN to a pre-existing MPLS baseline. A basic MPLS configuration is provided in case your network does not already have MPLS deployed.

To support MPLS-based VPNs the underlying MPLS baseline must provide the following functionality:

- Core-facing and loopback interfaces operational with MPLS family support
- An interior gateway protocol such as OSPF or IS-IS to provide reachability between the loopback addresses of the provider (P and PE) devices
- An MPLS signalling protocol such as LDP or RSVP to signal LSPs
- LSPs established between PE device loopback addresses

LSPs are needed between each pair of PE devices that participate in a given VPN. Its a good idea to build LSPs between all PE devices to accommodate future VPN growth. You configure LSPs at the `[edit protocols mpls]` hierarchy level. Unlike an MPLS configuration for a circuit cross-connect (CCC) connection, you do not need to manually associate the LSP with the PE device’s customer-facing (edge) interface. Instead, Layer 3 VPNs use BGP signaling to advertise site reachability. This BGP signaling automates the mapping of remote VPN sites to LSP forwarding next hops. This means that with a Layer 3 VPN explicit mapping of an LSP to a PE device’s edge-facing interface is not required.

Overview and Topology

Layer 3 VPNs allow customers to leverage the service provider's technical expertise to ensure efficient site-to-site routing. The customer edge (CE) device typically uses a routing protocol, such as BGP or OSPF, to exchange routes with the service provider edge (PE) device. Static routing is supported for Layer 3 VPNs, but a dynamic routing protocol is generally preferred.

Definition of a VPN involves changes to the local and remote PE devices only. No additional configuration is needed on the provider devices (assuming they already have a working MPLS baseline), because these devices only provide basic MPLS switching functions. The CE devices do not use MPLS and require only a basic interface and routing protocol configuration so they can interact with the PE devices.

In a Layer 3 VPN you configure the CE devices to peer with the local PE device. This is in contrast to a Layer 2 VPN, where the CE devices peer to each other as if they were on a shared link, despite their being connected through an MPLS-based provider core.

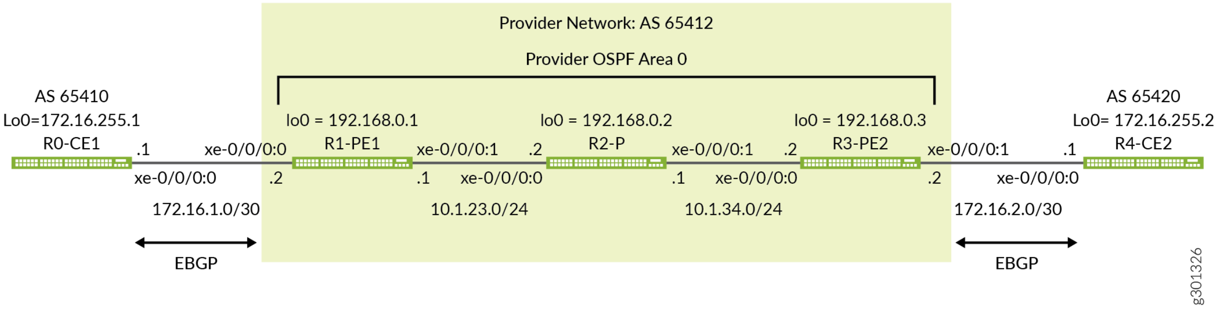
Once an MPLS baseline is in place, you must configure the following functionality on the PE devices to establish your MPLS-based Layer 3 VPN:

- A BGP group with `family inet-vpn unicast` support
- A routing instance with instance type `vrf` and a routing protocol definition that is compatible with the attached CE device
- The customer-facing interfaces on the PE devices configured with `family inet` along with an IPv4 address that places the interface on the same subnet as the attached CE device. If desired VLAN encapsulation and a corresponding VLAN ID can also be configured.

For proper end-to-end connectivity the CE device needs to be configured with a compatible IP subnet and routing protocol parameters to support peering with the PE device.

Figure 27 on page 260 shows the topology used in this example. The figure details the interface names, IP addressing, and routing protocols used in the provider and customer networks. It also highlights the peering relationship between the CE and PE devices. In this example you expect each CE device to form an EBGP peering session *to* the local PE device. Note that the provider network and both customer sites have an assigned autonomous system number to support BGP operation. In this example routing policy is applied at the CE devices to have them advertise the direct routes for their provider facing and loopback interfaces.

Figure 27: An MPLS-Based Layer 3 VPN with EBGP as the PE-CE Routing Protocol



Quick Configurations

IN THIS SECTION

Use the configurations in this section to quickly get your MPLS-based Layer 3 VPN up and running. The configurations include a functional MPLS baseline to support your Layer 3 VPN. This example focuses on the VPN aspects of the configuration. Refer to the following links for additional information on the baseline MPLS functionality used in this example:

- Configuring MPLS on Provider Edge EX8200 and EX4500 Switches Using Circuit Cross-Connect
- Configuring MPLS on EX8200 and EX4500 Provider Switches

CLI Quick Configuration

NOTE: The device configurations omit the management interface, static routes, system logging, system services, and user login information. These parts of the configuration vary by location and are not directly related to MPLS or VPN functionality.

Edit the following commands as needed for the specifics of your environment and paste them into the local CE (CE1) device terminal window when in configuration mode at the `[edit]` hierarchy:

The complete configuration for the CE1 device.

```
set system host-name ce1
set interfaces xe-0/0/0:0 description "Link from CE1 to PE1 for L3vpn"
set interfaces xe-0/0/0:0 unit 0 family inet address 172.16.1.1/30
set interfaces lo0 unit 0 family inet address 172.16.255.1/32
set routing-options router-id 172.16.255.1
set routing-options autonomous-system 65410
set protocols bgp group PE1 type external
set protocols bgp group PE1 export adv_direct
set protocols bgp group PE1 peer-as 65412
set protocols bgp group PE1 neighbor 172.16.1.2
set policy-options policy-statement adv_direct term 1 from protocol direct
set policy-options policy-statement adv_direct term 1 from route-filter 172.16.0.0/16 orlonger
set policy-options policy-statement adv_direct term 1 then accept
```

The complete configuration for PE1 device.

```
set system host-name pe1
set interfaces xe-0/0/0:0 description "Link from PE1 to CE1 for L3vpn"
set interfaces xe-0/0/0:0 unit 0 family inet address 172.16.1.2/30
set interfaces xe-0/0/0:1 description "Link from PE1 to p-router"
set interfaces xe-0/0/0:1 mtu 4000
set interfaces xe-0/0/0:1 unit 0 family inet address 10.1.23.1/24
set interfaces xe-0/0/0:1 unit 0 family mpls
set interfaces lo0 unit 0 family inet address 192.168.0.1/32
set routing-instances CE1_L3vpn protocols bgp group CE1 type external
set routing-instances CE1_L3vpn protocols bgp group CE1 peer-as 65410
set routing-instances CE1_L3vpn protocols bgp group CE1 neighbor 172.16.1.1
set routing-instances CE1_L3vpn instance-type vrf
set routing-instances CE1_L3vpn interface xe-0/0/0:0.0
set routing-instances CE1_L3vpn route-distinguisher 192.168.0.1:12
set routing-instances CE1_L3vpn vrf-target target:65412:12
set routing-options router-id 192.168.0.1
set routing-options autonomous-system 65412
set protocols bgp group ibgp type internal
set protocols bgp group ibgp local-address 192.168.0.1
set protocols bgp group ibgp family inet-vpn unicast
set protocols bgp group ibgp neighbor 192.168.0.3
set protocols mpls label-switched-path lsp_to_pe2 to 192.168.0.3
set protocols mpls interface xe-0/0/0:1.0
```

```
set protocols ospf traffic-engineering
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set protocols ospf area 0.0.0.0 interface xe-0/0/0:1.0
set protocols rsvp interface lo0.0
set protocols rsvp interface xe-0/0/0:1.0
```

The complete configuration for the P device.

```
set system host-name p
set interfaces xe-0/0/0:0 description "Link from p-router to PE1"
set interfaces xe-0/0/0:0 mtu 4000
set interfaces xe-0/0/0:0 unit 0 family inet address 10.1.23.2/24
set interfaces xe-0/0/0:0 unit 0 family mpls
set interfaces xe-0/0/0:1 description "Link from p-router to PE2"
set interfaces xe-0/0/0:1 mtu 4000
set interfaces xe-0/0/0:1 unit 0 family inet address 10.1.34.1/24
set interfaces xe-0/0/0:1 unit 0 family mpls
set interfaces lo0 unit 0 family inet address 192.168.0.2/32
set protocols mpls interface xe-0/0/0:0.0
set protocols mpls interface xe-0/0/0:1.0
set protocols ospf traffic-engineering
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set protocols ospf area 0.0.0.0 interface xe-0/0/0:0.0
set protocols ospf area 0.0.0.0 interface xe-0/0/0:1.0
set protocols rsvp interface lo0.0
set protocols rsvp interface xe-0/0/0:0.0
set protocols rsvp interface xe-0/0/0:1.0
```

The complete configuration for the PE2 device.

```
set system host-name pe2
set interfaces xe-0/0/0:1 description "Link from PE2 to CE2 for L3vpn"
set interfaces xe-0/0/0:1 unit 0 family inet address 172.16.2.2/30
set interfaces xe-0/0/0:0 description "Link from PE2 to p-router"
set interfaces xe-0/0/0:0 mtu 4000
set interfaces xe-0/0/0:0 unit 0 family inet address 10.1.34.2/24
set interfaces xe-0/0/0:0 unit 0 family mpls
set interfaces lo0 unit 0 family inet address 192.168.0.3/32
set routing-instances CE2_L3vpn protocols bgp group CE2 type external
set routing-instances CE2_L3vpn protocols bgp group CE2 peer-as 65420
set routing-instances CE2_L3vpn protocols bgp group CE2 neighbor 172.16.2.1
```

```
set routing-instances CE2_L3vpn instance-type vrf
set routing-instances CE2_L3vpn interface xe-0/0/0:1.0
set routing-instances CE2_L3vpn route-distinguisher 192.168.0.3:12
set routing-instances CE2_L3vpn vrf-target target:65412:12
set routing-options router-id 192.168.0.3
set routing-options autonomous-system 65412
set protocols bgp group ibgp type internal
set protocols bgp group ibgp local-address 192.168.0.3
set protocols bgp group ibgp family inet-vpn unicast
set protocols bgp group ibgp neighbor 192.168.0.1
set protocols mpls label-switched-path lsp_to_pe1 to 192.168.0.1
set protocols mpls interface xe-0/0/0:0.0
set protocols ospf traffic-engineering
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set protocols ospf area 0.0.0.0 interface xe-0/0/0:0.0
set protocols rsvp interface lo0.0
set protocols rsvp interface xe-0/0/0:0.0
```

The complete configuration for the CE2 device.

```
set system host-name ce2
set interfaces xe-0/0/0:0 description "Link from CE2 to PE2 for L3vpn"
set interfaces xe-0/0/0:0 unit 0 family inet address 172.16.2.1/30
set interfaces lo0 unit 0 family inet address 172.16.255.2/32
set routing-options router-id 172.16.255.2
set routing-options autonomous-system 65420
set protocols bgp group PE2 type external
set protocols bgp group PE2 export adv_direct
set protocols bgp group PE2 peer-as 65412
set protocols bgp group PE2 neighbor 172.16.2.2
set policy-options policy-statement adv_direct term 1 from protocol direct
set policy-options policy-statement adv_direct term 1 from route-filter 172.16.0.0/16 orlonger
set policy-options policy-statement adv_direct term 1 then accept
```

Be sure to commit the configuration changes on all devices when satisfied with your work. Congratulations on your new MPLS-based Layer 3 VPN! Refer to the "Verification" on page 277 section for the steps needed to confirm your Layer 3 VPN is working as expected.

Configure the Local PE (PE1) Device for a MPLS-Based Layer 3 VPN

IN THIS SECTION

This section covers the steps needed to configure the PE1 device for this example. The focus in on the PE devices because that is where the VPN configuration is housed. Refer to the "Quick Configurations" on page 260 section for the CE device and P device configurations used in this example.

Configure the MPLS Baseline (if needed)

Before you configure a Layer 3 VPN make sure the PE device has a working MPLS baseline. If you already having an MPLS baseline you can skip to the step-by-step procedure to add the Layer 3 VPN to the PE devices.

- Configure the hostname.

```
[edit]
user@pe1# set system host-name pe1
```

- Configure the core and loopback interfaces:

```
[edit]
user@pe1# set interfaces xe-0/0/0:1 description "Link from PE1 to P-router"
[edit]
user@pe1# set interfaces xe-0/0/0:1 mtu 4000
[edit]
user@pe1# set interfaces xe-0/0/0:1 unit 0 family inet address 10.1.23.1/24
[edit]
user@pe1# set interfaces xe-0/0/0:1 unit 0 family mpls
[edit]
user@pe1# set interfaces lo0 unit 0 family inet address 192.168.0.1/32
```

BEST PRACTICE: While a Layer 3 VPN can perform fragmentation at the ingress PE, its best practice to design the network so the CE can send a maximum sized frame without needing fragmentation. To ensure fragmentation does not occur the provider network should support the largest frame that the CE devices can generate *after* the MPLS and virtual routing and forwarding (VRF) labels are added by the PE device. This example leaves the CE devices at the default 1500-byte maximum transmission unit (MTU) while configuring the provider core to support a 4000 byte MTU. This ensures the CE devices cannot exceed the MTU in the provider's network even with the MPLS and VRF encapsulation overhead.

- Configure the protocols:

NOTE: Traffic engineering is supported for RSVP-signaled LSPs but is not required for basic MPLS switching or VPN deployment. The provided MPLS baseline uses RSVP to signal LSPs, and enables traffic engineering for OSPF. However, no path constraints are configured so you expect the LSPs to be routed over the interior gateway protocol's shortest path.

```
[edit]
user@pe1# set protocols ospf area 0.0.0.0 interface lo0.0 passive
[edit]
user@pe1# set protocols ospf area 0.0.0.0 interface xe-0/0/0:1.0
[edit]
user@pe1# set protocols ospf traffic-engineering
[edit]
user@pe1# set protocols mpls interface xe-0/0/0:1.0
[edit]
user@pe1# set protocols rsvp interface lo0.0
[edit]
user@pe1# set protocols rsvp interface xe-0/0/0:1.0
```

- Define the LSP to the remote PE device's loopback address:

```
[edit]
user@pe1#  set protocols mpls label-switched-path lsp_to_pe2 to 192.168.0.3
```

The MPLS baseline is now configured on the PE1 device. Keep going to configure the Layer 3 VPN.

Procedure

Step-by-Step Procedure

Follow these steps to configure the PE1 device for a Layer 3 VPN.

1. Configure the customer-facing interface:

   TIP: You can configure both an MPLS-based Layer 2 VPN and an MPLS-based Layer 3 VPN on the same PE device. However, you cannot configure the same customer edge-facing interface to support both a Layer 2 VPN and a Layer 3 VPN.

   ```
   [edit interfaces]
   user@pe1# set xe-0/0/0:0 description "Link from PE1 to CE1 for L3vpn"
   [edit]
   user@pe1# set xe-0/0/0:0 unit 0 family inet address 172.16.1.2/30
   ```

2. Configure a BGP group for the peering between the local and remote PE devices. Use the PE device's loopback address as the local address, and enable the `inet-vpn unicast` address family to support Layer 3 VPN route exchange. A routing policy for BGP is not needed on the PE devices in this example. By default the PE devices readvertise into IBGP the routes they learn over their EBGP peering to the CE device.

   ```
   [edit protocols bgp]
   user@pe1# set group ibgp local-address 192.168.0.1
   [edit protocols bgp]
   user@pe1# set group ibgp family inet-vpn unicast
   ```

   TIP: You can specify other address families if the PE to PE IBGP session needs to support non-VPN route exchange, such as regular IPv4 or IPv6 routes using the `inet` or `inet6` families, respectively.

3. Configure the BGP group type as internal.

   ```
   [edit protocols bgp]
   user@pe1# set group ibgp type internal
   ```

4. Configure the remote PE device's loopback address as a BGP neighbor.

```
[edit protocols bgp]
user@pe1# set group ibgp neighbor 192.168.0.3
```

5. Configure the router ID to match its loopback address, and define the BGP autonomous system number needed for BGP peering.

```
[edit routing-options]
user@pe1# set router-id 192.168.0.1
[edit routing-options]
user@pe1# set autonomous-system 65412
```

6. Configure the routing instance. Specify an instance name of *CE1_L3vpn*, and configure an `instance-type` of `vrf`.

```
[edit routing-instances]
user@pe1# set CE1_L3vpn instance-type vrf
```

7. Configure the PE device's customer-facing interface to belong to the routing instance.

```
[edit routing-instances]
user@pe1# set CE1_L3vpn interface xe-0/0/0:0.0
```

8. Configure the routing instance's route distinguisher. This setting is used to distinguish the routes sent from a particular VRF on a particular PE device. It should be unique for each routing instance on each PE device.

```
[edit routing-instances]
user@pe1# set CE1_L3vpn route-distinguisher 192.168.0.1:12
```

9. Configure the instance's virtual routing and forwarding (VRF) table route target. The `vrf-target` statement adds the specified community tag to all advertised routes while automatically matching

the same value for route import. Configuring matching route targets on the PE devices that share a given VPN is required for proper route exchange.

```
[edit routing-instances]
user@pe1# set CE1_L3vpn vrf-target target:65142:12
```

NOTE: You can create more complex policies by explicitly configuring VRF import and export policies using the import and export options. See *vrf-import* and *vrf-export* for details.

10. Configure the routing instance to support EBGP peering to the CE1 device. Direct interface peering to the CE1 end of the VRF link is used, and CE1's autonomous system number is correctly specified with the `peer-as` parameter.

```
[edit routing-instances]
user@pe1# set CE1_L3vpn protocols bgp group CE1 type external
[edit routing-instances]
user@pe1# set CE1_L3vpn protocols bgp group CE1 peer-as 65410
[edit routing-instances]
user@pe1# set CE1_L3vpn protocols bgp group CE1 neighbor 172.16.1.1
```

11. Commit your changes at the PE1 device and return to CLI operational mode.

```
[edit]
user@pe1# commit and-quit
```

Results

Display the results of the configuration on the PE1 device. The output reflects only the functional configuration added in this example.

```
user@pe1> show configuration
```

```
interfaces {
    xe-0/0/0:0 {
        description "Link from PE1 to CE1 for L3vpn";
```

```
        unit 0 {
             family inet {
        }
    }
    unit 0 {
        family inet {
            address 10.1.23.1/24;
        }
        family mpls;
    }
    ge-0/0/1 {
        description "Link from PE1 to P-router";
        mtu 4000;
        unit 0 {
            family inet {
                address 10.1.23.1/24;
            }
            family mpls;
        }
    }
    lo0 {
        unit 0 {
            family inet {
                address 192.168.0.1/32;
            }
        }
    }
}
routing-instances {
    CE1_L3vpn {
        protocols {
            bgp {
                group CE1 {
                    type external;
                    peer-as 65410;
                    neighbor 172.16.1.1;
                }
            }
        }
        instance-type vrf;
        interface xe-0/0/0:0.0;
        route-distinguisher 192.168.0.1:12;
        vrf-target target:65412:12;
```

```
    }
}
routing-options {
    router-id 192.168.0.1;
    autonomous-system 65412;
}
protocols {
    bgp {
        group ibgp {
            type internal;
            local-address 192.168.0.1;
            family inet-vpn {
                unicast;
            }
            neighbor 192.168.0.3;
        }
    }
    mpls {
        label-switched-path lsp_to_pe2 {
            to 192.168.0.3;
        }
        interface xe-0/0/0:1.0;
    }
    ospf {
        traffic-engineering;
        area 0.0.0.0 {
            interface lo0.0 {
                passive;
            }
            interface xe-0/0/0:1.0;
        }
    }
    rsvp {
        interface lo0.0;
        interface xe-0/0/0:1.0;
    }
}
```

Configure the Remote PE (PE2) Device for a MPLS-Based Layer 3 VPN

IN THIS SECTION

This section covers the steps needed to configure the PE1 device for this example. The focus in on the PE devices because that is where the VPN configuration is housed. Refer to the "Quick Configurations" on page 260 section for the CE device and P device configurations used in this example.

Configure the MPLS Baseline (if needed)

Before you configure a Layer 3 VPN make sure the PE device has a working MPLS baseline. If you already having an MPLS baseline you can skip to the step-by-step procedure to add the Layer 3 VPN to the PE devices.

- Configure the hostname.

```
[edit]
user@pe2# set system host-name pe2
```

- Configure the core and loopback interfaces:

```
[edit]
user@pe2# set interfaces xe-0/0/0:0 description "Link from PE2 to P-router"
[edit]
user@pe2# set interfaces xe-0/0/0:0 mtu 4000
[edit]
user@pe2# set interfaces xe-0/0/0:0 unit 0 family inet address 10.1.34.2/24
[edit]
user@pe2# set interfaces xe-0/0/0:0 unit 0 family mpls
[edit]
user@pe2# set interfaces lo0 unit 0 family inet address 192.168.0.3/32
```

BEST PRACTICE: While a Layer 3 VPN can perform fragmentation at the ingress PE, its best practice to design the network so the CE can send a maximum sized frame without needing fragmentation. To ensure fragmentation does not occur the provider network should support the largest frame that the CE devices can generate *after* the MPLS and virtual routing and forwarding (VRF) labels are added by the PE device. This example leaves the CE devices at the default 1500-byte maximum transmission unit (MTU) while configuring the provider core to support a 4000 byte MTU. This ensures the CE devices cannot exceed the MTU in the provider's network even with the MPLS and VRF encapsulation overhead.

- Configure the protocols:

NOTE: Traffic engineering is supported for RSVP-signaled LSPs but is not required for basic MPLS switching or VPN deployment. The provided MPLS baseline uses RSVP to signal LSPs, and enables traffic engineering for OSPF. However, no path constraints are configured so you expect the LSPs to be routed over the interior gateway protocol's shortest path.

```
[edit]
user@pe2# set protocols ospf area 0.0.0.0 interface lo0.0 passive
[edit]
user@pe2# set protocols ospf area 0.0.0.0 interface xe-0/0/0:0.0
[edit]
user@pe2# set protocols ospf traffic-engineering
[edit]
user@pe2# set protocols mpls interface xe-0/0/0:0.0
[edit]
user@pe2# set protocols rsvp interface lo0.0
[edit]
user@pe2# set protocols rsvp interface xe-0/0/0:0.0
```

- Define the LSP to the remote PE device's loopback address:

```
[edit]
user@pe2# set protocols mpls label-switched-path lsp_to_pe1 to 192.168.0.1
```

The MPLS baseline is now configured on the PE1 device. Keep going to configure the Layer 3 VPN.

Procedure

Step-by-Step Procedure

Follow these steps to configure the PE2 device for a Layer 3 VPN.

1. Configure the customer-facing interface:

   TIP: You can configure both an MPLS-based Layer 2 VPN and an MPLS-based Layer 3 VPN on the same PE device. However, you cannot configure the same customer edge-facing interface to support both a Layer 2 VPN and a Layer 3 VPN.

   ```
   [edit interfaces]
   user@pe2# set xe-0/0/0:1 description "Link from PE2 to CE2 for L3vpn"
   [edit]
   user@pe2# set xe-0/0/0:1 unit 0 family inet address 172.16.2.2/30
   ```

2. Configure a BGP group for the peering between the local and remote PE devices. Use the PE device's loopback address as the local address, and enable the `inet-vpn unicast` address family to support Layer 3 VPN route exchange.

   ```
   [edit protocols bgp]
   user@pe1# set group ibgp local-address 192.168.0.1
   [edit protocols bgp]
   user@pe1# set group ibgp family inet-vpn unicast
   ```

   TIP: You can specify other address families if the PE to PE IBGP session needs to support non-VPN route exchange, such as regular IPv4 or IPv6 routes using the `inet` or `inet6` families, respectively.

3. Configure the BGP group type as internal.

   ```
   [edit protocols bgp]
   user@pe2# set group ibgp type internal
   ```

4. Configure the PE1 device’s loopback address as a BGP neighbor.

```
[edit protocols bgp]
user@pe2# set group ibgp neighbor 192.168.0.1
```

5. Configure the router ID to match its loopback address, and define the BGP autonomous system number.

```
[edit routing-options]
user@pe2# set routing-options router-id 192.168.0.3
[edit routing-options]
user@pe2# set autonomous-system 65412
```

6. Configure the routing instance. Specify an instance name of *CE2_L3vpn*, with an `instance-type` of `vrf`.

```
[edit routing-instances]
user@pe2# set CE2_L3vpn instance-type vrf
```

7. Configure the PE device’s customer-facing interface to belong to the routing instance.

```
[edit routing-instances]
user@pe2# set CE2_L3vpn interface xe-0/0/0:1.0
```

8. Configure the routing instance’s route distinguisher. This setting is used to distinguish the routes sent from a particular VRF on a particular PE device. It should be unique for each routing instance on each PE device.

```
[edit routing-instances]
user@pe2# set CE2_L3vpn route-distinguisher 192.168.0.3:12
```

9. Configure the instance’s virtual routing and forwarding (VRF) table route target. The `vrf-target` statement adds the specified community tag to all advertised routes while automatically matching

the same value for route import. Configuring matching route targets on the PE devices that share a given VPN is required for proper route exchange.

```
[edit routing-instances]
user@pe2# set CE2_L3vpn vrf-target target:65412:12
```

NOTE: You can create more complex policies by explicitly configuring VRF import and export policies using the import and export options. See *vrf-import* and *vrf-export* for details.

10. Configure the routing instance to support EBGP peering to the CE2 device. Direct interface peering to the CE2 end of the VRF link is used, and CE2's autonomous system number is correctly specified with the `peer-as` parameter.

```
[edit routing-instances]
user@pe2# set CE2_L3vpn protocols bgp group CE2 type external
[edit routing-instances]
user@pe2# set CE2_L3vpn protocols bgp group CE2 peer-as 65420
[edit routing-instances]
user@pe2# set CE2_L3vpn protocols bgp group CE2 neighbor 172.16.2.1
```

11. Commit your changes at the PE2 device and return to CLI operational mode.

```
[edit]
user@pe2# commit and-quit
```

Results

Display the results of the configuration on the PE2 device. The output reflects only the functional configuration added in this example.

```
user@pe2> show configuration
```

```
interfaces {
    xe-0/0/0:0 {
        description "Link from PE2 to p-router";
```

```
            mtu 4000;
            unit 0 {
                family inet {
                    address 10.1.34.2/24;
                }
                family mpls;
            }
        }
        xe-0/0/0:1 {
            description "Link from PE2 to CE2 for L3vpn";
            unit 0 {
                family inet {
                    address 172.16.2.2/30;
                }
            }
        }
        lo0 {
            unit 0 {
                family inet {
                    address 192.168.0.3/32;
                }
            }
        }
    }
    routing-instances {
        CE2_L3vpn {
            protocols {
                bgp {
                    group CE2 {
                        type external;
                        peer-as 65420;
                        neighbor 172.16.2.1;
                    }
                }
            }
            instance-type vrf;
            interface xe-0/0/0:1.0;
            route-distinguisher 192.168.0.3:12;
            vrf-target target:65412:12;
        }
    }
    routing-options {
        router-id 192.168.0.3;
```

```
    autonomous-system 65412;
}
protocols {
    bgp {
        group ibgp {
            type internal;
            local-address 192.168.0.3;
            family inet-vpn {
                unicast;
            }
            neighbor 192.168.0.1;
        }
    }
    mpls {
        label-switched-path lsp_to_pe1 {
            to 192.168.0.1;
        }
        interface xe-0/0/0:0.0;
    }
    ospf {
        traffic-engineering;
        area 0.0.0.0 {
            interface lo0.0 {
                passive;
            }
            interface xe-0/0/0:0.0;
        }
    }
    rsvp {
        interface lo0.0;
        interface xe-0/0/0:0.0;
    }
}
```

Verification

IN THIS SECTION

Perform these tasks to verify that the MPLS-based Layer 3 VPN works properly:

Verify Provider OSPF Adjacencies and Route Exchange

Purpose

Confirm the OSPF protocol is working properly in the provider network by verifying adjacency status and OSPF learned routes to the loopback addresses of the remote provider devices. Proper IGP operation is critical for the successful establishment of MPLS LSPs.

Action

```
user@pe1> show ospf neighbor

Address          Interface              State           ID               Pri  Dead
10.1.23.2        xe-0/0/0:1.0           Full            192.168.0.2      128    37
```

```
user@pe1> show route protocol ospf | match 192.168

192.168.0.2/32     *[OSPF/10] 1w1d 23:59:43, metric 1
192.168.0.3/32     *[OSPF/10] 1w1d 23:59:38, metric 2
```

Meaning

The output shows that the PE1 device has established an OSPF adjacency to the P device (`192.168.0.2`). It also shows that the P and remote PE device loopback addresses (`192.168.0.2`) and (`192.168.0.3`) are correctly learned via OSPF at the local PE device.

Verify MPLS and RSVP Interface Settings

Purpose

Verify that the RSVP and MPLS protocols are configured to operate on the PE device's core-facing interface. This step also verifies that `family mpls` is correctly configured at the unit level of the PE device's core-facing interface.

Action

```
user@pe1> show mpls interface

Interface        State       Administrative groups (x: extended)
xe-0/0/0:1.0     Up         <none>
```

```
user@pe1> show rsvp interface

RSVP interface: 2 active
                          Active  Subscr- Static      Available   Reserved    Highwater
Interface          State  resv    iption  BW          BW          BW          mark
lo0.0                  Up       0   100%  0bps        0bps        0bps        0bps
xe-0/0/0:1.0           Up       1   100%  10Gbps      10Gbps      0bps        0bps
```

Meaning

The output shows that MPLS and RSVP are correctly configured on the local PE device's core and loopback interfaces.

Verify RSVP Signaled LSPs

Purpose

Verify that RSVP signaled ingress and egress LSPs are correctly established between the PE device's loopback addresses.

Action

```
user@pe1> show rsvp session

Ingress RSVP: 1 sessions
To              From            State   Rt Style Labelin Labelout LSPname
192.168.0.3     192.168.0.1     Up       0  1 FF       -       17 lsp_to_pe2
Total 1 displayed, Up 1, Down 0

Egress RSVP: 1 sessions
To              From            State   Rt Style Labelin Labelout LSPname
192.168.0.1     192.168.0.3     Up       0  1 FF       3        - lsp_to_pe1
Total 1 displayed, Up 1, Down 0

Transit RSVP: 0 sessions
Total 0 displayed, Up 0, Down 0
```

Meaning

The output shows that both the ingress and egress RSVP sessions are correctly established between the PE devices. Successful LSP establishment indicates the MPLS baseline is operational.

Verify BGP Session Status

Purpose

Verify that the IBGP session between the PE devices is correctly established with support for Layer 3 VPN network layer reachability information (NLRI). In this step you also confirm the local PE to CE EBGP session is established and correctly configured to exchange IPv4 routes.

Action

```
user@pe1> show bgp summary

Groups: 2 Peers: 2 Down peers: 0
Table          Tot Paths  Act Paths Suppressed    History Damp State    Pending
inet.0
                       0          0          0          0          0          0
bgp.l3vpn.0
                       2          2          0          0          0          0
```

```
Peer                     AS      InPkt     OutPkt    OutQ   Flaps Last Up/Dwn State|#Active/
Received/Accepted/Damped...
172.16.1.1            65410      26038      25938       0       0 1w1d 3:12:32 Establ
  CE1_L3vpn.inet.0: 1/2/2/0
192.168.0.3           65412         19         17       0       0        6:18 Establ
  CE1_L3vpn.inet.0: 2/2/2/0
  bgp.l3vpn.0: 2/2/2/0
```

Meaning

The output shows the IBGP session to the remote PE device (`192.168.0.3`) has been correctly established (`Establ`), and through the `Up/Dwn` field, how long the session has been in the current state (`6:18`). The `flaps` field confirms that no state transitions have occurred (`0`), indicating the session is stable. Also note that Layer 3 VPN routes (NLRI) have been learned from the remote PE as shown by the presence of a `bgp.l3vpn.0` table.

The display also confirms the EBGP session to the local CE1 device (`172.16.1.1`) is established and that IPv4 routes have been received from the CE1 device and installed in the CE1 device routing instance (`CE1_L3vpn.inet.0`)

This output confirms that the BGP peering between the PE devices, and to the CE device, is working properly to support your Layer 3 VPN.

Verify Layer 3 VPN Routes in the Routing Table

Purpose

Confirm that the routing table on the PE1 device is populated with Layer 3 VPN routes advertised by the remote PE. These routes are used to forward traffic to the remote CE device.

Action

```
user@pe1> show route table bgp.l3vpn.0

bgp.l3vpn.0: 2 destinations, 2 routes (2 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

192.168.0.3:12:172.16.2.0/30
                   *[BGP/170] 00:22:45, localpref 100, from 192.168.0.3
                      AS path: I, validation-state: unverified
                    >  to 10.1.23.2 via xe-0/0/0:1.0, label-switched-path lsp_to_pe2
```

```
192.168.0.3:12:172.16.255.2/32
                   *[BGP/170] 00:22:43, localpref 100, from 192.168.0.3
                      AS path: 65420 I, validation-state: unverified
                    >  to 10.1.23.2 via xe-0/0/0:1.0, label-switched-path lsp_to_pe2
```

```
user@pe1> show route table CE1_L3vpn.inet.0

CE1_L3vpn.inet.0: 5 destinations, 6 routes (5 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

172.16.1.0/30      *[Direct/0] 1w1d 03:29:44
                    >  via xe-0/0/0:0.0
                    [BGP/170] 1w1d 03:29:41, localpref 100
                      AS path: 65410 I, validation-state: unverified
                    >  to 172.16.1.1 via xe-0/0/0:0.0
172.16.1.2/32      *[Local/0] 1w1d 03:29:44
                       Local via xe-0/0/0:0.0
172.16.2.0/30      *[BGP/170] 00:23:28, localpref 100, from 192.168.0.3
                      AS path: I, validation-state: unverified
                    >  to 10.1.23.2 via xe-0/0/0:1.0, label-switched-path lsp_to_pe2
172.16.255.1/32    *[BGP/170] 1w1d 03:29:41, localpref 100
                      AS path: 65410 I, validation-state: unverified
                    >  to 172.16.1.1 via xe-0/0/0:0.0
172.16.255.2/32    *[BGP/170] 00:23:26, localpref 100, from 192.168.0.3
                      AS path: 65420 I, validation-state: unverified
                    >  to 10.1.23.2 via xe-0/0/0:1.0, label-switched-path lsp_to_pe2
```

Meaning

The `show route table bgp.l3vpn.0` command displays the Layer 3 VPN routes that have been received from the remote PE device. The `show route table CE1_L3vpn.inet.0` command lists the all routes that have been imported into the `CE1_L3vpn` routing instance. These entries represent the routes learned from the local EBGP peering to the CE1 device, in addition to those routes received from the remote PE2 device with a matching route target.

Both tables show the remote Layer 3 VPN routes are correctly associated with the `lsp_to_pe2` LSP as a forwarding next hop. The outputs confirm the local PE device has learned the routes associated with the remote CE2 location from the PE2 device. It also shows that the local PE will forward Layer 3 VPN traffic to the remote PE2 device using MPLS transport over the provider network.

Ping the Remote PE Device Using the Layer 3 VPN Connection

Purpose

Verify Layer 3 VPN connectivity between the local and remote PE devices using ping. This command verifies the Layer 3 VPN routing and MPLS forwarding operation between the PE devices.

Action

```
user@pe1> ping routing-instance CE1_L3vpn 172.16.2.2 source 172.16.1.2 count 2

PING 172.16.2.2 (172.16.2.2): 56 data bytes
64 bytes from 172.16.2.2: icmp_seq=0 ttl=61 time=128.235 ms
64 bytes from 172.16.2.2: icmp_seq=1 ttl=61 time=87.597 ms

--- 172.16.2.2 ping statistics ---
2 packets transmitted, 2 packets received, 0% packet loss
round-trip min/avg/max/stddev = 87.597/107.916/128.235/20.319 m
```

Meaning

The output confirms that the Layer 3 VPN control and forwarding planes are operating correctly between the PE devices.

Verify End-to-End Operation of the CE Devices Over the Layer 3 VPN

Purpose

Verify Layer 3 VPN connectivity between the CE devices. This step confirms the CE devices have operational interfaces and are properly configured for EBGP based Layer 3 connectivity. This is done by verifying the local CE1 device has learned the remote CE device’s routes, and by confirming the CE devices are able to pass traffic end-to-end between their loopback addresses.

Action

```
user@ce1> show route protocol bgp

inet.0: 12 destinations, 12 routes (12 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both
```

```
172.16.2.0/30      *[BGP/170] 00:40:50, localpref 100
                      AS path: 65412 I, validation-state: unverified
                    >  to 172.16.1.2 via xe-0/0/0:0.0
172.16.255.2/32    *[BGP/170] 00:40:49, localpref 100
                      AS path: 65412 65420 I, validation-state: unverified
                    >  to 172.16.1.2 via xe-0/0/0:0.0

inet6.0: 1 destinations, 1 routes (1 active, 0 holddown, 0 hidden)
```

```
user@ce1> ping 172.16.255.2 source 172.16.255.1 size 1472 do-not-fragment count 2

PING 172.16.255.2 (172.16.255.2): 1472 data bytes
1480 bytes from 172.16.255.2: icmp_seq=0 ttl=61 time=79.245 ms
1480 bytes from 172.16.255.2: icmp_seq=1 ttl=61 time=89.125 ms

--- 172.16.255.2 ping statistics ---
2 packets transmitted, 2 packets received, 0% packet loss
round-trip min/avg/max/stddev = 79.245/84.185/89.125/4.940 ms
```

Meaning

The output shows that Layer 3 VPN based connectivity is working correctly between the CE devices. The local CE device has learned the remote CE device's VRF interface and loopback routes through BGP. The ping is generated to the loopback address of the remote CE device, and is sourced from the local CE device's loopback address using the `source 172.16.255.1` argument. Adding the `do-not-fragment` and `size 1472` switches confirms that the CE devices are able to pass 1500-byte IP packets without evoking fragmentation in the local PE device.

NOTE: The `size 1472` argument added to the `ping` command generates 1472 bytes of echo data. An additional 8 bytes of Internet Control Message Protocol (ICMP) and 20 bytes of IP header are added to bring the total payload size to 1500-bytes. Adding the `do-not-fragment` switch ensures that the local CE and PE devices cannot perform fragmentation. This ping method confirms that fragmentation is not needed when exchanging standard 1500-byte maximum length Ethernet frames between the CE devices.

These results confirm the MPLS-based Layer 3 VPN is working correctly.

RELATED DOCUMENTATION

Example: Configuring MPLS on EX8200 and EX4500 Switches

Example: Configure MPLS-Based Layer 2 VPNs

IPv6 Traffic over Layer 3 VPNs

IN THIS SECTION

Understanding IPv6 Layer 3 VPNs

The interfaces between the PE and CE routers of a Layer 3 VPN can be configured to carry IP version 6 (IPv6) traffic. IP allows numerous nodes on different networks to interoperate seamlessly. IPv4 is currently used in intranets and private networks, as well as the Internet. IPv6 is the successor to IPv4, and is based for the most part on IPv4.

In the Juniper Networks implementation of IPv6, the service provider implements an MPLS-enabled IPv4 backbone to provide VPN service for IPv6 customers. The PE routers have both IPv4 and IPv6 capabilities. They maintain IPv6 VPN routing and forwarding (VRF) tables for their IPv6 sites and encapsulate IPv6 traffic in MPLS frames that are then sent into the MPLS core network. IPv6 VPN routing over MPLS is also known as 6VPE.

IPv6 for Layer 3 VPNs is supported for BGP and for static routes.

IPv6 over Layer 3 VPNs is described in RFC 4659, *BGP-MPLS IP Virtual Private Network (VPN) Extension for IPv6 VPN.*

Configuring Layer 3 VPNs to Carry IPv6 Traffic

IN THIS SECTION

You can configure IP version 6 (IPv6) between the PE and CE routers of a Layer 3 VPN. The PE router must have the PE router to PE router BGP session configured with the `family inet6-vpn` statement. The CE router must be capable of receiving IPv6 traffic. You can configure BGP or static routes between the PE and CE routers.

The following sections explains how to configure IPv6 VPNs between the PE routers:

Configuring IPv6 on the PE Router

To configure IPv6 between the PE and CE routers, include the `family inet6-vpn` statement in the configuration on the PE router:

```
family inet6-vpn {
    (any | multicast | unicast) {
        aggregate-label community community-name;
        prefix-limit maximum prefix-limit;
        rib-group rib-group-name;
    }
}
```

For a list of hierarchy levels at which you can configure this statement, see the statement summary section for this statement.

You also must include the `ipv6-tunneling` statement:

```
ipv6-tunneling;
```

You can include this statement at the following hierarchy levels:

- `[edit protocols mpls]`
- `[edit logical-systems logical-system-name protocols mpls]`

Configuring the Connection Between the PE and CE Routers

To support IPv6 routes, you must configure BGP, OSPF version 3, IS-IS, or static routes for the connection between the PE and CE routers in the Layer 3 VPN. You can configure BGP to handle just IPv6 routes or both IP version 4 (IPv4) and IPv6 routes.

For more information about IS-IS see Example: Configuring IS-IS,

The following sections explain how to configure BGP and static routes:

Configuring BGP on the PE Router to Handle IPv6 Routes

To configure BGP in the Layer 3 VPN routing instance to handle IPv6 routes, include the `bgp` statement:

```
bgp {
    group group-name {
        local-address IPv6-address;
        family inet6 {
            unicast;
        }
        peer-as as-number;
        neighbor IPv6-address;
    }
}
```

You can include this statement at the following hierarchy levels:

- `[edit routing-instances routing-instance-name protocols]`
- `[edit logical-systems logical-system-name routing-instances routing-instance-name protocols]`

Configuring BGP on the PE Router for IPv4 and IPv6 Routes

To configure BGP in the Layer 3 VPN routing instance to handle both IPv4 and IPv6 routes, include the `bgp` statement:

```
bgp {
    group group-name {
        local-address IPv4-address;
        family inet {
            unicast;
        }
```

```
        family inet6 {
            unicast;
        }
        peer-as as-number;
        neighbor address;
    }
}
```

You can include this statement at the following hierarchy levels:

- `[edit routing-instances routing-instance-name protocols]`
- `[edit logical-systems logical-system-name routing-instances routing-instance-name protocols]`

NOTE: The `[edit logical-systems]` hierarchy level is not applicable in ACX Series routers.

Configuring OSPF Version 3 on the PE Router

To configure OSPF version 3 in the Layer 3 VPN routing instance to handle IPv6 routes, include the `ospf3` statement:

```
ospf3 {
    area area-id {
        interface interface-name;
    }
}
```

You can include this statement at the following hierarchy levels:

- `[edit routing-instances routing-instance-name protocols]`
- `[edit logical-systems logical-system-name routing-instances routing-instance-name protocols]`

NOTE: The `[edit logical-systems]` hierarchy level is not applicable in ACX Series routers.

Configuring Static Routes on the PE Router

To configure a static route to the CE router in the Layer 3 VPN routing instance, include the `routing-options` statement:

```
routing-options {
    rib routing-table.inet6.0 {
        static {
            defaults {
                static-options;
            }
        }
    }
}
```

You can include this statement at the following hierarchy levels:

- `[edit routing-instances routing-instance-name]`
- `[edit logical-systems logical-system-name routing-instances routing-instance-name]`

NOTE: The `[edit logical-systems]` hierarchy level is not applicable in ACX Series routers.

Configuring IPv6 on the Interfaces

You need to configure IPv6 on the PE router interfaces to the CE routers and on the CE router interfaces to the PE routers.

To configure the interface to handle IPv6 routes, include the `family inet6` statement:

```
family inet6 {
    address ipv6-address;
}
```

You can include this statement at the following hierarchy levels:

- `[edit interfaces interface-name unit unit-number]`
- `[edit logical-systems logical-system-name interfaces interface-name unit unit-number]`

NOTE: The `[edit logical-systems]` hierarchy level is not applicable in ACX Series routers.

If you have configured the Layer 3 VPN to handle both IPv4 and IPv6 routes, configure the interface to handle both IPv4 and IPv6 routes by including the `unit` statement:

```
unit unit-number {
    family inet {
        address ipv4-address;
    }
    family inet6 {
        address ipv6-address;
    }
}
```

You can include this statement at the following hierarchy levels:

- `[edit interfaces interface-name]`
- `[edit logical-systems logical-system-name interfaces interface-name]`

NOTE: The `[edit logical-systems]` hierarchy level is not applicable in ACX Series routers.

Example: Tunneling Layer 3 VPN IPv6 Islands over an IPv4 Core Using IBGP and Independent Domains

IN THIS SECTION

This example shows how to configure Junos OS to tunnel IPv6 over a Layer 3 VPN IPv4 network. Internal BGP (IBGP) is used between the customer edge (CE) and provider edge (PE) devices, as described in Internet draft draft-marques-ppvpn-ibgp-version.txt, *RFC2547bis networks using internal BGP as PE-CE protocol*, instead of the more typical external BGP (EBGP) PE-CE connections.

Requirements

No special configuration beyond device initialization is required before you configure this example.

All PE routers participating in a Layer 3 VPN with the `independent-domain` statement in its configuration must be running Junos OS Release 6.3 or later.

Overview

This example shows one method of enabling a router to participate in a customer VPN autonomous-system (AS) domain and to transparently exchange routing information through a Layer 3 VPN without the customer network attributes being visible to the carrier network, and without the carrier network attributes being visible to the customer network.

As an added requirement, the customer network in this example is based on IPv6, while the provider network uses IPv4.

The `independent-domain` feature is useful when customer route attributes need to be transparently forwarded across the VPN network without even the service-provider (SP) AS path appearing in the routes. In a typical Layer 3 VPN, the route attributes such as the originator ID, cluster list, route metric, and AS path are not transparent from one CE device to another CE device.

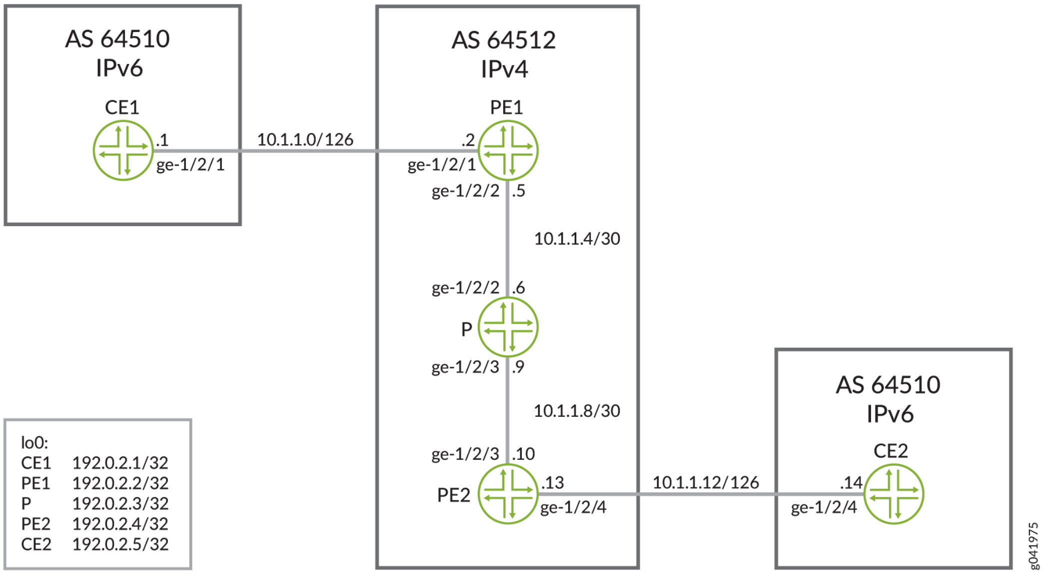
For example, suppose you have a customer VRF whose AS is 1. The customer advertises routes to you through BGP (either IBGP or EBGP). Your core network (the primary routing instance) uses AS 3. Without `independent-domain` configured, if the customer advertises 10.0.0.0/24 to you through BGP, the prefix contains the customer's AS 1 in the AS path. To transport the advertisement across the core to the other PE devices, your core AS 3 is added to the AS path by multiprotocol BGP (MP-BGP). The AS path is now 3 1. When the prefix is advertised out of the core back into the Layer 3 VPN at a remote PE device, the Layer 3 VPN AS 1 is added again, making the AS Path 1 3 1, which is an AS loop. The `independent-domain` statement ensures that only the ASs in the routing-instance are checked during loop detection, and the main, primary routing instances (your core's AS 3) is not considered. This is done by using the attribute 128 (attribute set), which is an optional transitive attribute. The attribute set hides the route's AS path, local preference, and so on, so that those do not appear during the loop check.

NOTE: In Junos OS 10.4 and later, you can specify the `no-attrset` option of `independent-domain` so that instead of using attribute 128 (attribute set), Junos OS simply does loop checking on routing-instance ASs without considering your core's AS used in MP-BGP. This is useful if you are using the `local-as` feature, and you only want to configure independent domains to maintain the independence of local ASs in the routing instance, and perform BGP loop detection only for the specified local ASs in the routing instance. In this case, you can disable the attribute set message.

Topology

Figure 28 on page 292 shows the sample network.

Figure 28: Layer 3 VPN IPv6 Islands over an IPv4 Core Using IBGP and Independent Domains



"CLI Quick Configuration" on page 293 shows the configuration for all of the devices in Figure 28 on page 292.

The section "Configuring Device PE1" on page 295 describes the steps on Device PE1.

Configuration

IN THIS SECTION

CLI Quick Configuration

To quickly configure this example, copy the following commands, paste them into a text file, remove any line breaks, change any details necessary to match your network configuration, and then copy and paste the commands into the CLI at the `[edit]` hierarchy level.

Device CE1

```
set interfaces ge-1/2/1 unit 0 family inet6 address ::10.1.1.1/126
set interfaces ge-1/2/1 unit 0 family mpls
set interfaces lo0 unit 0 family inet6 address ::192.0.2.1/32
set protocols bgp group toPE1 type internal
set protocols bgp group toPE1 family inet6 unicast
set protocols bgp group toPE1 export send-direct
set protocols bgp group toPE1 neighbor ::10.1.1.2
set policy-options policy-statement send-direct from protocol direct
set policy-options policy-statement send-direct then accept
set routing-options router-id 192.0.2.1
set routing-options autonomous-system 64510
```

Device CE2

```
set interfaces ge-1/2/4 unit 0 family inet6 address ::10.1.1.14/126
set interfaces ge-1/2/4 unit 0 family mpls
set interfaces lo0 unit 0 family inet6 address ::192.0.2.5/32
set protocols bgp group toPE2 type internal
set protocols bgp group toPE2 family inet6 unicast
set protocols bgp group toPE2 export send-direct
set protocols bgp group toPE2 neighbor ::10.1.1.13
set policy-options policy-statement send-direct from protocol direct
set policy-options policy-statement send-direct then accept
```

```
set routing-options router-id 192.0.2.5
set routing-options autonomous-system 64510
```

Device PE1

```
set interfaces ge-1/2/1 unit 0 family inet6 address ::10.1.1.2/126
set interfaces ge-1/2/2 unit 0 family inet address 10.1.1.5/30
set interfaces ge-1/2/2 unit 0 family mpls
set interfaces lo0 unit 0 family inet address 192.0.2.2/32
set protocols mpls ipv6-tunneling
set protocols mpls interface ge-1/2/2.0
set protocols bgp group toPE2 type internal
set protocols bgp group toPE2 local-address 192.0.2.2
set protocols bgp group toPE2 family inet6-vpn unicast
set protocols bgp group toPE2 neighbor 192.0.2.4
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set protocols ospf area 0.0.0.0 interface ge-1/2/2.0
set protocols ldp interface ge-1/2/2.0
set protocols ldp interface lo0.0
set routing-instances red instance-type vrf
set routing-instances red interface ge-1/2/1.0
set routing-instances red route-distinguisher 64512:1
set routing-instances red vrf-target target:64512:1
set routing-instances red routing-options router-id 192.0.2.2
set routing-instances red routing-options autonomous-system 64510
set routing-instances red routing-options autonomous-system independent-domain
set routing-instances red protocols bgp group toCE1 type internal
set routing-instances red protocols bgp group toCE1 family inet6 unicast
set routing-instances red protocols bgp group toCE1 neighbor ::10.1.1.1
set routing-options router-id 192.0.2.2
set routing-options autonomous-system 64512
```

Device P

```
set interfaces ge-1/2/2 unit 0 family inet address 10.1.1.6/30
set interfaces ge-1/2/2 unit 0 family mpls
set interfaces ge-1/2/3 unit 0 family inet address 10.1.1.9/30
set interfaces ge-1/2/3 unit 0 family mpls
set interfaces lo0 unit 0 family inet address 192.0.2.3/32
set protocols mpls interface all
set protocols ospf area 0.0.0.0 interface lo0.0 passive
```

```
set protocols ospf area 0.0.0.0 interface all
set protocols ldp interface all
set routing-options router-id 192.0.2.3
```

Device PE2

```
set interfaces ge-1/2/3 unit 0 family inet address 10.1.1.10/30
set interfaces ge-1/2/3 unit 0 family mpls
set interfaces ge-1/2/4 unit 0 family inet6 address ::10.1.1.13/126
set interfaces lo0 unit 0 family inet address 192.0.2.4/32
set protocols mpls ipv6-tunneling
set protocols mpls interface ge-1/2/3.0
set protocols bgp group toPE1 type internal
set protocols bgp group toPE1 local-address 192.0.2.4
set protocols bgp group toPE1 family inet6-vpn unicast
set protocols bgp group toPE1 neighbor 192.0.2.2
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set protocols ospf area 0.0.0.0 interface ge-1/2/3.0
set protocols ldp interface ge-1/2/3.0
set protocols ldp interface lo0.0
set routing-instances red instance-type vrf
set routing-instances red interface ge-1/2/4.0
set routing-instances red route-distinguisher 64512:1
set routing-instances red vrf-target target:64512:1
set routing-instances red routing-options router-id 192.0.2.4
set routing-instances red routing-options autonomous-system 64510
set routing-instances red routing-options autonomous-system independent-domain
set routing-instances red protocols bgp group toCE2 type internal
set routing-instances red protocols bgp group toCE2 family inet6 unicast
set routing-instances red protocols bgp group toCE2 neighbor ::10.1.1.14
set routing-options router-id 192.0.2.4
set routing-options autonomous-system 64512
```

Configuring Device PE1

Step-by-Step Procedure

The following example requires you to navigate various levels in the configuration hierarchy. For information about navigating the CLI, see *Using the CLI Editor in Configuration Mode* in the CLI User Guide.

To configure Device PE1:

1. Configure the interfaces.

```
[edit interfaces]
user@PE1# set ge-1/2/1 unit 0 family inet6 address ::10.1.1.2/126
user@PE1# set ge-1/2/2 unit 0 family inet address 10.1.1.5/30
user@PE1# set ge-1/2/2 unit 0 family mpls
user@PE1# set lo0 unit 0 family inet address 192.0.2.2/32
```

2. Configure MPLS on the interfaces.

```
[edit protocols mpls]
user@PE1# set ipv6-tunneling
user@PE1# set interface ge-1/2/2.0
```

3. Configure BGP.

```
[edit protocols bgp group toPE2]
user@PE1# set type internal
user@PE1# set local-address 192.0.2.2
user@PE1# set family inet6-vpn unicast
user@PE1# set neighbor 192.0.2.4
```

4. Configure an interior gateway protocol (IGP).

```
[edit protocols ospf area 0.0.0.0]
user@PE1# set interface lo0.0 passive
user@PE1# set interface ge-1/2/2.0
```

5. Configure a signaling protocol.

```
[edit protocols]
user@PE1# set ldp interface ge-1/2/2.0
user@PE1# set ldp interface lo0.0
```

6. Configure the routing instance.

```
[edit routing-instances red]
user@PE1# set instance-type vrf
user@PE1# set interface ge-1/2/1.0
user@PE1# set route-distinguisher 64512:1
user@PE1# set vrf-target target:64512:1
user@PE1# set routing-options router-id 192.0.2.2
user@PE1# set protocols bgp group toCE1 type internal
user@PE1# set protocols bgp group toCE1 family inet6 unicast
user@PE1# set protocols bgp group toCE1 neighbor ::10.1.1.1
```

7. In the routing instance, include the AS number of the customer network, and include the `independent-domain` statement.

```
[edit routing-instances red routing-options]
user@PE1# set autonomous-system 64510
user@PE1# set autonomous-system independent-domain
```

8. In the main instance, configure the router ID and the provider AS number.

```
[edit routing-options]
user@PE1# set router-id 192.0.2.2
user@PE1# set autonomous-system 64512
```

Results

From configuration mode, confirm your configuration by entering the `show interfaces`, `show protocols`, `show routing-instances`, and `show routing-options` commands. If the output does not display the intended configuration, repeat the instructions in this example to correct the configuration.

```
user@PE1# show interfaces
interfaces {
    ge-1/2/1 {
        unit 0 {
            family inet6 {
                address ::10.1.1.2/126;
            }
```

```
            }
        }
        ge-1/2/2 {
            unit 0 {
                family inet {
                    address 10.1.1.5/30;
                }
                family mpls;
            }
        }
        lo0 {
            unit 0 {
                family inet {
                    address 192.0.2.2/32;
                }
            }
        }
    }
```

```
user@PE1# show protocols
mpls {
    ipv6-tunneling;
    interface ge-1/2/2.0;
}
bgp {
    group toPE2 {
        type internal;
        local-address 192.0.2.2;
        family inet6-vpn {
            unicast;
        }
        neighbor 192.0.2.4;
    }
}
ospf {
    area 0.0.0.0 {
        interface lo0.0 {
            passive;
        }
        interface ge-1/2/2.0;
    }
```

```
}
ldp {
    interface ge-1/2/2.0;
    interface lo0.0;
}
```

```
user@PE1# show routing-instances
red {
    instance-type vrf;
    interface ge-1/2/1.0;
    route-distinguisher 64512:1;
    vrf-target target:64512:1;
    routing-options {
        router-id 192.0.2.2;
        autonomous-system 64510 independent-domain;
    }
    protocols {
        bgp {
            group toCE1 {
                type internal;
                family inet6 {
                    unicast;
                }
                neighbor ::10.1.1.1;
            }
        }
    }
}
```

```
user@PE1# show routing-options
router-id 192.0.2.2;
autonomous-system 64512;
```

If you are done configuring the device, enter `commit` from configuration mode.

Verification

IN THIS SECTION

Confirm that the configuration is working properly.

Verifying That the CE Devices Have Connectivity

Purpose

Make sure that the tunnel is operating.

Action

From operational mode, enter the `ping` command.

```
user@CE1> ping ::192.0.2.5

PING6(56=40+8+8 bytes) ::10.1.1.1 --> ::192.0.2.5
16 bytes from ::192.0.2.5, icmp_seq=0 hlim=63 time=1.943 ms
16 bytes from ::192.0.2.5, icmp_seq=1 hlim=63 time=1.587 ms
^C
--- ::192.0.2.5 ping6 statistics ---
2 packets transmitted, 2 packets received, 0% packet loss
round-trip min/avg/max/std-dev = 1.587/1.765/1.943/0.178 ms
```

```
user@CE2> ping ::192.0.2.1
PING6(56=40+8+8 bytes) ::10.1.1.14 --> ::192.0.2.1
16 bytes from ::192.0.2.1, icmp_seq=0 hlim=63 time=2.097 ms
16 bytes from ::192.0.2.1, icmp_seq=1 hlim=63 time=1.610 ms
^C
--- ::192.0.2.1 ping6 statistics ---
```

```
2 packets transmitted, 2 packets received, 0% packet loss
round-trip min/avg/max/std-dev = 1.610/1.853/2.097/0.244 ms
```

Meaning

The IPv6 CE devices can communicate over the core IPv4 network.

Checking the AS Paths

Purpose

Make sure that the provider AS number does not appear in the CE device routing tables.

Action

From operational mode, enter the `show route protocol bgp detail` command.

```
user@CE1> show route protocol bgp detail

inet6.0: 8 destinations, 8 routes (8 active, 0 holddown, 0 hidden)
::192.0.2.5/32 (1 entry, 1 announced)
        *BGP    Preference: 170/-101
                Next hop type: Indirect
                Address: 0x9514354
                Next-hop reference count: 6
                Source: ::10.1.1.2
                Next hop type: Router, Next hop index: 924
                Next hop: ::10.1.1.2 via ge-1/2/1.0, selected
                Session Id: 0x500001
                Protocol next hop: ::10.1.1.2
                Indirect next hop: 0x971c000 262147 INH Session ID: 0x500002
                State: <Active Int Ext>
                Local AS: 64510 Peer AS: 64510
                Age: 50:58      Metric2: 0
                Validation State: unverified
                Task: BGP_64510.::10.1.1.2+45824
                Announcement bits (2): 0-KRT 2-Resolve tree 2
                AS path: I
                Accepted
                Localpref: 100
                Router ID: 192.0.2.2
```

```
::10.1.1.12/126 (1 entry, 1 announced)
        *BGP    Preference: 170/-101
                Next hop type: Indirect
                Address: 0x9514354
                Next-hop reference count: 6
                Source: ::10.1.1.2
                Next hop type: Router, Next hop index: 924
                Next hop: ::10.1.1.2 via ge-1/2/1.0, selected
                Session Id: 0x500001
                Protocol next hop: ::10.1.1.2
                Indirect next hop: 0x971c000 262147 INH Session ID: 0x500002
                State: <Active Int Ext>
                Local AS: 64510 Peer AS: 64510
                Age: 50:58      Metric2: 0
                Validation State: unverified
                Task: BGP_64510.::10.1.1.2+45824
                Announcement bits (2): 0-KRT 2-Resolve tree 2
                AS path: I
                Accepted
                Localpref: 100
                Router ID: 192.0.2.2
```

```
user@CE2> show route protocol bgp detail


inet6.0: 8 destinations, 8 routes (8 active, 0 holddown, 0 hidden)
::192.0.2.1/32 (1 entry, 1 announced)
        *BGP    Preference: 170/-101
                Next hop type: Indirect
                Address: 0x9514354
                Next-hop reference count: 6
                Source: ::10.1.1.13
                Next hop type: Router, Next hop index: 914
                Next hop: ::10.1.1.13 via ge-1/2/4.0, selected
                Session Id: 0x400001
                Protocol next hop: ::10.1.1.13
                Indirect next hop: 0x971c000 262150 INH Session ID: 0x400002
                State: <Active Int Ext>
                Local AS: 64510 Peer AS: 64510
                Age: 50:41      Metric2: 0
```

```
                  Validation State: unverified
                  Task: BGP_64510.::10.1.1.13+59329
                  Announcement bits (2): 0-KRT 2-Resolve tree 2
                  AS path: I
                  Accepted
                  Localpref: 100
                  Router ID: 192.0.2.4

::10.1.1.0/126 (1 entry, 1 announced)
        *BGP      Preference: 170/-101
                  Next hop type: Indirect
                  Address: 0x9514354
                  Next-hop reference count: 6
                  Source: ::10.1.1.13
                  Next hop type: Router, Next hop index: 914
                  Next hop: ::10.1.1.13 via ge-1/2/4.0, selected
                  Session Id: 0x400001
                  Protocol next hop: ::10.1.1.13
                  Indirect next hop: 0x971c000 262150 INH Session ID: 0x400002
                  State: <Active Int Ext>
                  Local AS: 64510 Peer AS: 64510
                  Age: 50:41      Metric2: 0
                  Validation State: unverified
                  Task: BGP_64510.::10.1.1.13+59329
                  Announcement bits (2): 0-KRT 2-Resolve tree 2
                  AS path: I
                  Accepted
                  Localpref: 100
                  Router ID: 192.0.2.4
```

Meaning

The output shows that for the BGP routes on the CE devices, the AS path attribute does not include the provider AS 64512.

SEE ALSO

Configuring the Ingress Router for MPLS-Signaled LSPs

Minimum RSVP Configuration

RELATED DOCUMENTATION

Routing Policies, Firewall Filters, and Traffic Policers User Guide

Junos OS Routing Protocols Library

Configuring an AS for Layer 3 VPNs

IN THIS SECTION

Configuring Layer 3 VPNs to Carry IBGP Traffic

An independent AS domain is separate from the primary routing instance domain. An AS is a set of routers that are under a single technical administration and that generally use a single IGP and metrics to propagate routing information within the set of routers. An AS appears to other ASs to have a single, coherent interior routing plan and presents a consistent picture of what destinations are reachable through it.

Configuring an independent domain allows you to keep the AS paths of the independent domain from being shared with the AS path and AS path attributes of other domains, including the master routing instance domain.

If you are using BGP on the router, you must configure an AS number.

When you configure BGP as the routing protocol between a PE router and a CE router in a Layer 3 VPN, you typically configure external peering sessions between the Layer 3 VPN service provider and the customer network ASs.

If the customer network has several sites advertising routes through an external BGP session to the service provider network and if the same AS is used by all the customer sites, the CE routers reject routes from the other CE routers. They detect a loop in the BGP AS path attribute.

To prevent the CE routers from rejecting each other’s routes, you could configure the following:

- PE routers advertising routes received from remote PE routers can remap the customer network AS number to its own AS number.

- AS path loops can be configured.
- The customer network can be configured with different AS numbers at each site.

These types of configurations can work when there are no BGP routing exchanges between the customer network and other networks. However, they do have limitations for customer networks that use BGP internally for purposes other than carrying traffic between the CE routers and the PE routers. When those routes are advertised outside the customer network, the service provider ASs are present in the AS path.

To improve the transparency of Layer 3 VPN services for customer networks, you can configure the routing instance for the Layer 3 VPN to isolate the customer's network attributes from the service provider's network attributes.

When you include the `independent-domain` statement in the Layer 3 VPN routing instance configuration, BGP attributes received from the customer network (from the CE router) are stored in a BGP attribute (ATTRSET) that functions like a stack. When that route is advertised from the remote PE router to the remote CE router, the original BGP attributes are restored. This is the default behavior for BGP routes that are advertised to Layer 3 VPNs located in different domains.

This functionality is described in the Internet draft draft-marques-ppvpn-ibgp-version.txt, *RFC 2547bis Networks Using Internal BGP as PE-CE Protocol*.

To allow a Layer 3 VPN to transport IBGP traffic, include the `independent-domain` statement:

```
independent-domain;
```

You can include this statement at the following hierarchy levels:

- `[edit routing-instances` *`routing-instance-name`* `routing-options autonomous-system` *`number`*`]`
- `[edit logical-systems` *`logical-system-name`* `routing-instances` *`routing-instance-name`* `routing-options autonomous-system` *`number`*`]`

NOTE: All PE routers participating in a Layer 3 VPN with the `independent-domain` statement in its configuration must be running Junos OS Release 6.3 or later.

NOTE: The `[edit logical-systems]` hierarchy level is not applicable in ACX Series routers.

The independent domain uses the transitive path attribute 128 (attribute set) to tunnel the independent domain's BGP attributes through the Internal BGP (IBGP) core. In Junos OS Release 10.3 and later, if BGP receives attribute 128 and you have not configured an independent domain in any routing instance, BGP treats the received attribute 128 as an unknown attribute.

There is a limit of 16 ASs for each domain.

SEE ALSO

Example: Tunneling Layer 3 VPN IPv6 Islands over an IPv4 Core Using IBGP and Independent Domains | 290
Disabling Attribute Set Messages on Independent AS Domains for BGP Loop Detection

Example: Configuring a Layer 3 VPN with Route Reflection and AS Override

IN THIS SECTION

Suppose that you are a service provider providing a managed MPLS-based Layer 3 VPN service. Your customer has several sites and requires BGP routing to customer edge (CE) devices at each site.

Requirements

No special configuration beyond device initialization is required before configuring this example.

Overview

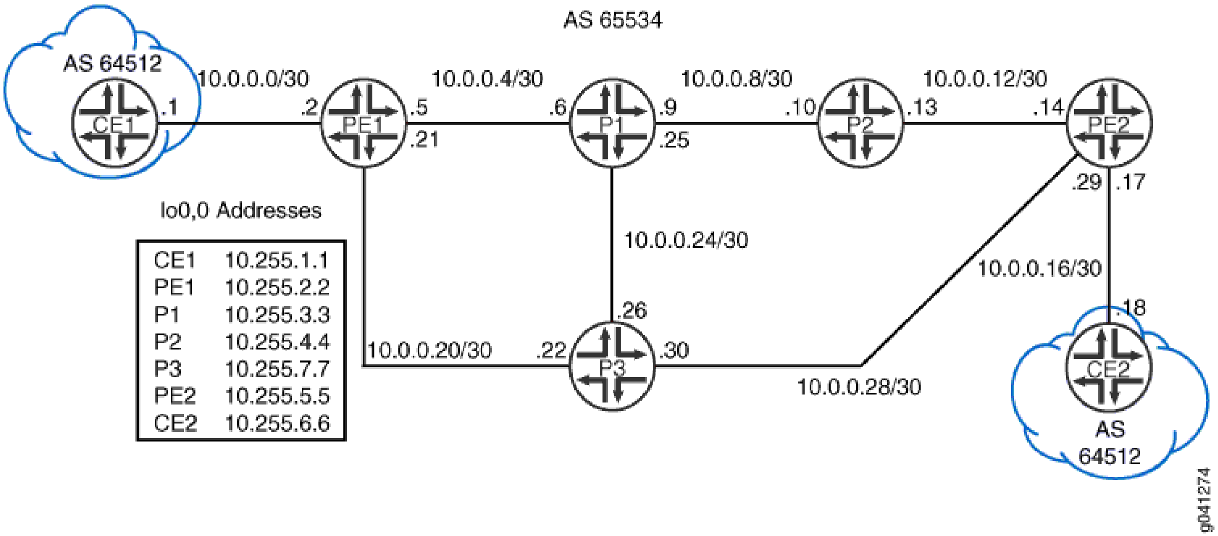
IN THIS SECTION

This example has two CE devices, two provider edge (PE) devices, and several provider core devices. The provider network is also using IS-IS to support LDP and BGP loopback reachability Device P2 is acting as a route reflector (RR). Both CE devices are in autonomous system (AS) 64512. The provider network is in AS 65534.

The `as-override` statement is applied to the PE devices, thus replacing the CE device's AS number with that of the PE device. This prevents the customer AS number from appearing more than once in the AS path attribute.

Figure 29 on page 307 shows the topology used in this example.

Figure 29: AS Override Topology



CE1	10.255.1.1
PE1	10.255.2.2
P1	10.255.3.3
P2	10.255.4.4
P3	10.255.7.7
PE2	10.255.5.5
CE2	10.255.6.6

"CLI Quick Configuration" on page 308 shows the configuration for all of the devices in Figure 29 on page 307. The section "Step-by-Step Procedure" on page 313 describes the steps on Device PE1.

Topology

Configuration

IN THIS SECTION

Procedure

CLI Quick Configuration

To quickly configure this example, copy the following commands, paste them into a text file, remove any line breaks, change any details necessary to match your network configuration, and then copy and paste the commands into the CLI at the `[edit]` hierarchy level.

Device CE1

```
set interfaces ge-1/2/0 unit 0 family inet address 10.0.0.1/30
set interfaces ge-1/2/0 unit 0 family iso
set interfaces lo0 unit 0 family inet address 10.255.1.1/32
set interfaces lo0 unit 0 family iso address 49.0001.0010.0000.0101.00
set protocols bgp group PE type external
set protocols bgp group PE family inet unicast
set protocols bgp group PE export ToBGP
set protocols bgp group PE peer-as 65534
set protocols bgp group PE neighbor 10.0.0.2
set policy-options policy-statement ToBGP term Direct from protocol direct
set policy-options policy-statement ToBGP term Direct then accept
set routing-options router-id 10.255.1.1
set routing-options autonomous-system 64512
```

Device P1

```
set interfaces ge-1/2/0 unit 0 family inet address 10.0.0.6/30
set interfaces ge-1/2/0 unit 0 family iso
set interfaces ge-1/2/0 unit 0 family mpls
set interfaces ge-1/2/1 unit 0 family inet address 10.0.0.9/30
set interfaces ge-1/2/1 unit 0 family iso
set interfaces ge-1/2/1 unit 0 family mpls
set interfaces ge-1/2/2 unit 0 family inet address 10.0.0.25/30
set interfaces ge-1/2/2 unit 0 family iso
set interfaces ge-1/2/2 unit 0 family mpls
set interfaces lo0 unit 0 family inet address 10.255.3.3/32
set interfaces lo0 unit 0 family iso address 49.0001.0010.0000.0303.00
set protocols mpls interface all
set protocols mpls interface fxp0.0 disable
set protocols bgp group l3vpn type internal
set protocols bgp group l3vpn local-address 10.255.3.3
set protocols bgp group l3vpn family inet-vpn unicast
```

```
set protocols bgp group l3vpn peer-as 65534
set protocols bgp group l3vpn local-as 65534
set protocols bgp group l3vpn neighbor 10.255.4.4
set protocols isis interface all level 2 metric 10
set protocols isis interface all level 1 disable
set protocols isis interface fxp0.0 disable
set protocols isis interface lo0.0 level 2 metric 0
set protocols ldp deaggregate
set protocols ldp interface all
set protocols ldp interface fxp0.0 disable
set routing-options router-id 10.255.3.3
```

Device P2

```
set interfaces ge-1/2/0 unit 0 family inet address 10.0.0.10/30
set interfaces ge-1/2/0 unit 0 family iso
set interfaces ge-1/2/0 unit 0 family mpls
set interfaces ge-1/2/1 unit 0 family inet address 10.0.0.13/30
set interfaces ge-1/2/1 unit 0 family iso
set interfaces ge-1/2/1 unit 0 family mpls
set interfaces lo0 unit 0 family inet address 10.255.4.4/32
set interfaces lo0 unit 0 family iso address 49.0001.0010.0000.0404.00
set protocols mpls interface all
set protocols mpls interface fxp0.0 disable
set protocols bgp group Core-RRClients type internal
set protocols bgp group Core-RRClients local-address 10.255.4.4
set protocols bgp group Core-RRClients family inet-vpn unicast
set protocols bgp group Core-RRClients cluster 10.255.4.4
set protocols bgp group Core-RRClients peer-as 65534
set protocols bgp group Core-RRClients neighbor 10.255.3.3
set protocols bgp group Core-RRClients neighbor 10.255.7.7
set protocols bgp group Core-RRClients neighbor 10.255.2.2
set protocols bgp group Core-RRClients neighbor 10.255.5.5
set protocols isis interface all level 2 metric 10
set protocols isis interface all level 1 disable
set protocols isis interface fxp0.0 disable
set protocols isis interface lo0.0 level 2 metric 0
set protocols ldp deaggregate
set protocols ldp interface all
set protocols ldp interface fxp0.0 disable
```

```
set routing-options router-id 10.255.4.4
set routing-options autonomous-system 65534
```

Device P3

```
set interfaces ge-1/2/0 unit 0 family inet address 10.0.0.22/30
set interfaces ge-1/2/0 unit 0 family iso
set interfaces ge-1/2/0 unit 0 family mpls
set interfaces ge-1/2/1 unit 0 family inet address 10.0.0.26/30
set interfaces ge-1/2/1 unit 0 family iso
set interfaces ge-1/2/1 unit 0 family mpls
set interfaces ge-1/2/2 unit 0 family inet address 10.0.0.30/30
set interfaces ge-1/2/2 unit 0 family iso
set interfaces ge-1/2/2 unit 0 family mpls
set interfaces lo0 unit 0 family inet address 10.255.7.7/32
set interfaces lo0 unit 0 family iso address 49.0001.0010.0000.0707.00
set protocols mpls interface all
set protocols mpls interface fxp0.0 disable
set protocols bgp group l3vpn type internal
set protocols bgp group l3vpn local-address 10.255.7.7
set protocols bgp group l3vpn family inet-vpn unicast
set protocols bgp group l3vpn peer-as 65534
set protocols bgp group l3vpn local-as 65534
set protocols bgp group l3vpn neighbor 10.255.4.4
set protocols isis interface all level 2 metric 10
set protocols isis interface all level 1 disable
set protocols isis interface fxp0.0 disable
set protocols isis interface lo0.0 level 2 metric 0
set protocols ldp deaggregate
set protocols ldp interface all
set protocols ldp interface fxp0.0 disable
set routing-options router-id 10.255.7.7
```

Device PE1

```
set interfaces ge-1/2/0 unit 0 family inet address 10.0.0.2/30
set interfaces ge-1/2/0 unit 0 family iso
set interfaces ge-1/2/0 unit 0 family mpls
set interfaces ge-1/2/1 unit 0 family inet address 10.0.0.5/30
set interfaces ge-1/2/1 unit 0 family iso
set interfaces ge-1/2/1 unit 0 family mpls
```

```
set interfaces ge-1/2/2 unit 0 family inet address 10.0.0.21/30
set interfaces ge-1/2/2 unit 0 family iso
set interfaces ge-1/2/2 unit 0 family mpls
set interfaces lo0 unit 0 family inet address 10.255.2.2/32
set interfaces lo0 unit 0 family iso address 49.0001.0010.0000.0202.00
set protocols mpls interface ge-1/2/2.0
set protocols mpls interface ge-1/2/1.0
set protocols mpls interface lo0.0
set protocols mpls interface fxp0.0 disable
set protocols bgp group l3vpn type internal
set protocols bgp group l3vpn local-address 10.255.2.2
set protocols bgp group l3vpn family inet-vpn unicast
set protocols bgp group l3vpn peer-as 65534
set protocols bgp group l3vpn local-as 65534
set protocols bgp group l3vpn neighbor 10.255.4.4
set protocols isis interface ge-1/2/1.0 level 2 metric 10
set protocols isis interface ge-1/2/1.0 level 1 disable
set protocols isis interface ge-1/2/2.0 level 2 metric 10
set protocols isis interface ge-1/2/2.0 level 1 disable
set protocols isis interface fxp0.0 disable
set protocols isis interface lo0.0 level 2 metric 0
set protocols ldp deaggregate
set protocols ldp interface ge-1/2/1.0
set protocols ldp interface ge-1/2/2.0
set protocols ldp interface fxp0.0 disable
set protocols ldp interface lo0.0
set routing-instances VPN-A instance-type vrf
set routing-instances VPN-A interface ge-1/2/0.0
set routing-instances VPN-A route-distinguisher 65534:1234
set routing-instances VPN-A vrf-target target:65534:1234
set routing-instances VPN-A protocols bgp group CE type external
set routing-instances VPN-A protocols bgp group CE family inet unicast
set routing-instances VPN-A protocols bgp group CE neighbor 10.0.0.1 peer-as 64512
set routing-instances VPN-A protocols bgp group CE neighbor 10.0.0.1 as-override
set routing-options router-id 10.255.2.2
set routing-options autonomous-system 65534
```

Device PE2

```
set interfaces ge-1/2/0 unit 0 family inet address 10.0.0.14/30
set interfaces ge-1/2/0 unit 0 family iso
set interfaces ge-1/2/0 unit 0 family mpls
```

```
set interfaces ge-1/2/1 unit 0 family inet address 10.0.0.17/30
set interfaces ge-1/2/1 unit 0 family iso
set interfaces ge-1/2/2 unit 0 family inet address 10.0.0.29/30
set interfaces ge-1/2/2 unit 0 family iso
set interfaces ge-1/2/2 unit 0 family mpls
set interfaces lo0 unit 0 family inet address 10.255.5.5/32
set interfaces lo0 unit 0 family iso address 49.0001.0010.0000.0505.00
set protocols mpls interface ge-1/2/0.0
set protocols mpls interface ge-1/2/2.0
set protocols mpls interface lo0.0
set protocols mpls interface fxp0.0 disable
set protocols bgp group l3vpn type internal
set protocols bgp group l3vpn local-address 10.255.5.5
set protocols bgp group l3vpn family inet-vpn unicast
set protocols bgp group l3vpn peer-as 65534
set protocols bgp group l3vpn local-as 65534
set protocols bgp group l3vpn neighbor 10.255.4.4
set protocols isis interface ge-1/2/0.0 level 2 metric 10
set protocols isis interface ge-1/2/0.0 level 1 disable
set protocols isis interface ge-1/2/2.0 level 2 metric 10
set protocols isis interface ge-1/2/2.0 level 1 disable
set protocols isis interface fxp0.0 disable
set protocols isis interface lo0.0 level 2 metric 0
set protocols ldp deaggregate
set protocols ldp interface ge-1/2/0.0
set protocols ldp interface ge-1/2/2.0
set protocols ldp interface fxp0.0 disable
set protocols ldp interface lo0.0
set routing-instances VPN-A instance-type vrf
set routing-instances VPN-A interface ge-1/2/1.0
set routing-instances VPN-A route-distinguisher 65534:1234
set routing-instances VPN-A vrf-target target:65534:1234
set routing-instances VPN-A protocols bgp group CE type external
set routing-instances VPN-A protocols bgp group CE family inet unicast
set routing-instances VPN-A protocols bgp group CE neighbor 10.0.0.18 peer-as 64512
set routing-instances VPN-A protocols bgp group CE neighbor 10.0.0.18 as-override
set routing-options router-id 10.255.5.5
set routing-options autonomous-system 65534
```

Device CE2

```
set interfaces ge-1/2/0 unit 0 family inet address 10.0.0.18/30
set interfaces ge-1/2/0 unit 0 family iso
set interfaces lo0 unit 0 family inet address 10.255.6.6/32
set interfaces lo0 unit 0 family iso address 49.0001.0010.0000.0606.00
set protocols bgp group PE type external
set protocols bgp group PE family inet unicast
set protocols bgp group PE export ToBGP
set protocols bgp group PE peer-as 65534
set protocols bgp group PE neighbor 10.0.0.17
set policy-options policy-statement ToBGP term Direct from protocol direct
set policy-options policy-statement ToBGP term Direct then accept
set routing-options router-id 10.255.6.6
set routing-options autonomous-system 64512
```

Step-by-Step Procedure

The following example requires you to navigate various levels in the configuration hierarchy. For information about navigating the CLI, see *Using the CLI Editor in Configuration Mode* in the CLI User Guide.

To configure AS override:

1. Configure the interfaces.

   To enable MPLS, include the protocol family on the interface so that the interface does not discard incoming MPLS traffic.

   ```
   [edit interfaces]
   user@PE1# set ge-1/2/0 unit 0 family inet address 10.0.0.2/30
   user@PE1# set ge-1/2/0 unit 0 family iso
   user@PE1# set ge-1/2/0 unit 0 family mpls
   user@PE1# set ge-1/2/1 unit 0 family inet address 10.0.0.5/30
   user@PE1# set ge-1/2/1 unit 0 family iso
   user@PE1# set ge-1/2/1 unit 0 family mpls
   user@PE1# set ge-1/2/2 unit 0 family inet address 10.0.0.21/30
   user@PE1# set ge-1/2/2 unit 0 family iso
   user@PE1# set ge-1/2/2 unit 0 family mpls
   user@PE1# set lo0 unit 0 family inet address 10.255.2.2/32
   user@PE1# set lo0 unit 0 family iso address 49.0001.0010.0000.0202.00
   ```

2. Add the interface to the MPLS protocol to establish the control plane level connectivity.

   Set up the IGP so that the provider devices can communicate with each other.

   To establish a mechanism to distribute MPLS labels, enable LDP. Optionally, for LDP, enable forwarding equivalence class (FEC) deaggregation, which results in faster global convergence.

```
[edit protocols]
user@PE1# set mpls interface ge-1/2/2.0
user@PE1# set mpls interface ge-1/2/1.0
user@PE1# set mpls interface lo0.0
user@PE1# set mpls interface fxp0.0 disable
user@PE1# set isis interface ge-1/2/1.0 level 2 metric 10
user@PE1# set isis interface ge-1/2/1.0 level 1 disable
user@PE1# set isis interface ge-1/2/2.0 level 2 metric 10
user@PE1# set isis interface ge-1/2/2.0 level 1 disable
user@PE1# set isis interface fxp0.0 disable
user@PE1# set isis interface lo0.0 level 2 metric 0
user@PE1# set ldp deaggregate
user@PE1# set ldp interface ge-1/2/1.0
user@PE1# set ldp interface ge-1/2/2.0
user@PE1# set ldp interface fxp0.0 disable
user@PE1# set ldp interface lo0.0
```

3. Enable the internal BGP (IBGP) connection to peer with the RR only, using the IPv4 VPN unicast address family.

```
[edit protocols bgp group l3vpn]
user@PE1# set type internal
user@PE1# set local-address 10.255.2.2
user@PE1# set family inet-vpn unicast
user@PE1# set peer-as 65534
user@PE1# set local-as 65534
user@PE1# set neighbor 10.255.4.4
```

4. Configure the routing instance, including the `as-override` statement.

Create the routing-Instance (VRF) on the PE device, setting up the BGP configuration to peer with Device CE1.

```
[edit routing-instances VPN-A]
user@PE1# set instance-type vrf
user@PE1# set interface ge-1/2/0.0
user@PE1# set route-distinguisher 65534:1234
user@PE1# set vrf-target target:65534:1234
user@PE1# set protocols bgp group CE type external
user@PE1# set protocols bgp group CE family inet unicast
user@PE1# set protocols bgp group CE neighbor 10.0.0.1 peer-as 64512
user@PE1# set protocols bgp group CE neighbor 10.0.0.1 as-override
```

5. Configure the router ID and the AS number.

```
[edit routing-options]
user@PE1# set router-id 10.255.2.2
user@PE1# set autonomous-system 65534
```

Results

From configuration mode, confirm your configuration by entering the `show interfaces`, `show protocols`, `show routing-instances`, and `show routing-options` commands. If the output does not display the intended configuration, repeat the configuration instructions in this example to correct it.

```
user@PE1# show interfaces
ge-1/2/0 {
    unit 2 {
        family inet {
            address 10.0.0.2/30;
        }
        family iso;
        family mpls;
    }
}
ge-1/2/1 {
    unit 5 {
        family inet {
            address 10.0.0.5/30;
```

```
        }
        family iso;
        family mpls;
    }
}
ge-1/2/2 {
    unit 21 {
        family inet {
            address 10.0.0.21/30;
        }
        family iso;
        family mpls;
    }
}
lo0 {
    unit 0 {
        family inet {
            address 10.255.2.2/32;
        }
        family iso {
            address 49.0001.0010.0000.0202.00;
        }
    }
}
```

```
user@PE1# show protocols
mpls {
    interface ge-1/2/2.0;
    interface ge-1/2/1.0;
    interface lo0.0;
    interface fxp0.0 {
        disable;
    }
}
bgp {
    group l3vpn {
        type internal;
        local-address 10.255.2.2;
        family inet-vpn {
            unicast;
        }
```

```
        peer-as 65534;
        local-as 65534;
        neighbor 10.255.4.4;
    }
}
isis {
    interface ge-1/2/1.0 {
        level 2 metric 10;
        level 1 disable;
    }
    interface ge-1/2/2.0 {
        level 2 metric 10;
        level 1 disable;
    }
    interface fxp0.0 {
        disable;
    }
    interface lo0.0 {
        level 2 metric 0;
    }
}
ldp {
    deaggregate;
    interface ge-1/2/1.0;
    interface ge-1/2/2.0;
    interface fxp0.0 {
        disable;
    }
    interface lo0.0;
}
```

```
user@PE1# show routing-instances
VPN-A {
    instance-type vrf;
    interface ge-1/2/0.0;
    route-distinguisher 65534:1234;
    vrf-target target:65534:1234;
    protocols {
        bgp {
            group CE {
                type external;
```

```
                family inet {
                    unicast;
                }
                neighbor 10.0.0.1 {
                    peer-as 64512;
                    as-override;
                }
            }
        }
    }
}
```

```
user@PE1# show routing-options
router-id 10.255.2.2;
autonomous-system 65534;
```

If you are done configuring the device, enter **commit** from configuration mode.

Verification

IN THIS SECTION

Confirm that the configuration is working properly.

Checking AS Path to the CE Devices

Purpose

Display information on Device PE1 about the AS path attribute for the route to Device CE2’s loopback interface.

Action

On Device PE1, from operational mode, enter the `show route table VPN-A.inet.0 10.255.6.6` command.

```
user@PE1> show route table VPN-A.inet.0 10.255.6.6



VPN-A.inet.0: 5 destinations, 6 routes (5 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

10.255.6.6/32      *[BGP/170] 02:19:35, localpref 100, from 10.255.4.4
                      AS path: 64512 I, validation-state: unverified
                    > to 10.0.0.22 via ge-1/2/2.0, Push 300032, Push 299776(top)
```

Meaning

The output shows that Device PE1 has an AS path for 10.255.6.6/32 as coming from AS 64512.

Checking How the Route to Device CE2 Is Advertised

Purpose

Make sure the route to Device CE2 is advertised to Device CE1 as if it is coming from the MPLS core.

Action

On Device PE1, from operational mode, enter the `show route advertising-protocol bgp 10.0.0.1` command.

```
user@PE1> show route advertising-protocol bgp 10.0.0.1

VPN-A.inet.0: 5 destinations, 6 routes (5 active, 0 holddown, 0 hidden)
  Prefix                  Nexthop              MED     Lclpref    AS path
* 10.0.0.16/30            Self                                    I
* 10.255.1.1/32           10.0.0.1                                65534 I
* 10.255.6.6/32           Self                                    65534 I
```

Meaning

The output indicates that Device PE1 is advertising only its own AS number in the AS path.

Checking the Route on Device CE1

Purpose

Make sure that Device CE1 contains only the provider AS number in the AS path for the route to Device CE2.

Action

From operational mode, enter the `show route table inet.0 terse 10.255.6.6` command.

```
user@CE1> show route table inet.0 terse 10.255.6.6

inet.0: 5 destinations, 6 routes (5 active, 0 holddown, 1 hidden)
+ = Active Route, - = Last Active, * = Both

A V Destination        P Prf   Metric 1   Metric 2  Next hop        AS path
* ? 10.255.6.6/32      B 170        100                             65534 65534 I
  unverified                                       >10.0.0.2
```

Meaning

The output indicates that Device CE1 has a route to Device CE2. The loop issue is resolved with the use of the `as-override` statement.

One route is hidden on the CE device. This is because Junos OS does not perform a BGP split horizon. Generally, split horizon in BGP is unnecessary, because any routes that might be received back by the originator are less preferred due to AS path length (for EBGP), AS path loop detection (IBGP), or other BGP metrics. Advertising routes back to the neighbor from which they were learned has a negligible effect on the router's performance, and is the correct thing to do.

SEE ALSO

Understanding AS Override

Configuring the Algorithm That Determines the Active Route to Evaluate AS Numbers in AS Paths for VPN Routes

By default, the third step of the algorithm that determines the active route evaluates the length of the AS path but not the contents of the AS path. In some VPN scenarios with BGP multiple path routes, it can also be useful to compare the AS numbers of the AS paths and to have the algorithm select the route whose AS numbers match.

To configure the algorithm that selects the active path to evaluate the AS numbers in AS paths for VPN routes:

- Include the `as-path-compare` statement at the `[edit routing-instances` *`routing-instance-name`* `routing-options multipath]` hierarchy level.

NOTE: The `as-path-compare` statement is not supported for the default routing instance.

Limiting VPN Routes Using Route Resolution

IN THIS SECTION

This topic describes limiting VPN routes by configuring route resolution on PE routers and route reflectors and by configuring the PE router to accept a limited number of prefix from a CE router.

Example: Configuring Route Resolution on PE Routers

IN THIS SECTION

This example shows how to configure a routing table to accept routes from specific routing tables. It also shows how to configure a routing table to use specific import policies to produce a route resolution table to resolve routes.

Requirements

Before you begin, configure a Layer 3 VPN, as shown in one of the following examples:

- "Example: Configuring Interprovider Layer 3 VPN Option A" on page 450
- "Example: Configuring Interprovider Layer 3 VPN Option B" on page 479

Overview

One method to achieve IPv4 route scaling is to modify how BGP routes are added to the forwarding tables. By default, the Routing Protocol Process (rpd) adds all the routes in inet.0 and inet.3 to the resolution tree. Normally, this includes the resolved IPv4 BGP routes, which can increase memory consumption. To achieve better scaling for IPv4 routes, this example shows how to configure the Junos OS so that resolved BGP routes are not added to the resolution tree. This is achieved by applying an import policy on the route resolution table, which ensures it does not accept any BGP routes from inet.0.

You would apply this configuration to all provider edge (PE) routers in the Layer 3 VPN.

Configuration

IN THIS SECTION

CLI Quick Configuration

To quickly configure this example, copy the following commands, paste them into a text file, remove any line breaks, change any details necessary to match your network configuration, and then copy and paste the commands into the CLI at the `[edit]` hierarchy level.

PE Router

```
set policy-options policy-statement protocol-bgp from protocol bgp
set policy-options policy-statement protocol-bgp then reject
set routing-options resolution rib inet.0 import protocol-bgp
```

Procedure

Step-by-Step Procedure

The following example requires you to navigate various levels in the configuration hierarchy. For information about navigating the CLI, see *Using the CLI Editor in Configuration Mode* in the CLI User Guide.

To configure route resolution:

1. Configure the routing policy.

   ```
   [edit policy-options policy-statement protocol-bgp]
   user@PE# set from protocol bgp
   user@PE# set then reject
   ```

2. Apply the routing policy.

   ```
   [edit routing-options resolution]
   user@PE# set rib inet.0 import protocol-bgp
   ```

3. If you are done configuring the device, commit the configuration.

   ```
   [edit]
   user@PE# commit
   ```

Results

Confirm your configuration by issuing the `show policy-options` and `show routing-options` commands.

```
user@PE# show policy-options
policy-statement protocol-bgp {
```

```
    from protocol bgp;
    then reject;
}
```

```
user@PE# show routing-options
resolution {
    rib inet.0 {
        import protocol-bgp;
    }
}
```

Verification

Confirm that the configuration is working properly by running the following commands:

- *show route*
- *show route forwarding-table*
- show route resolution

SEE ALSO

Example: Configuring Route Resolution on Route Reflectors | **324**

Example: Configuring Route Resolution on Route Reflectors

IN THIS SECTION

This example shows how to change the default resolution behavior on a route reflector (RR) to use inet.0 for next-hop resolution instead of inet.3.

Requirements

Before you begin, configure a Layer 3 VPN, as shown in one of the following examples:

- "Example: Configuring Interprovider Layer 3 VPN Option A" on page 450
- "Example: Configuring Interprovider Layer 3 VPN Option B" on page 479

Overview

One scenario for route resolution is when you have a label-switched path configured from an RR to a provider edge (PE) router, or when the PE routers only peer with the RR. This can result in routes being hidden. To resolve this issue, you can change the default resolution behavior to use inet.0 for next-hop resolution.

By default, the bgp.l3vpn.0 routing table stores all VPN-IPv4 unicast routes. This table is present on any router that has Layer 3 VPNs configured, including PE routers and RRs.

When a Layer 3 VPN router receives a route from another Layer 3 VPN router, it places the route into its bgp.l3vpn.0 routing table. The route is resolved using the information in the inet.3 routing table. This means that when BGP receives a route destined for table bgp.l3vpn.0, the protocol nexthop (received BGP nexthop) has its forwarding nexthop recursively determined from the inet.3 table. The resulting route is converted into IPv4 format and redistributed to all *routing-instance-name*.inet.0 routing tables if it matches the VRF import policy.

On an RR with no attached customer edge (CE) routers, the `resolution rib bgp.l3vpn.0 resolution-ribs inet.0` configuration causes routes in bgp.l3vpn.0 to use the information in inet.0 instead of inet.3 to resolve routes. You should not use this configuration on a router that is directly attached to a CE router. In other words, do not use `resolution rib bgp.l3vpn.0 resolution-ribs inet.0` on a PE router.

If you want both inet.0 and inet.3 to be used, you must configure both, as in `set resolution rib bgp.l3vpn.0 resolution-ribs [inet.0 inet.3]`.

In this example, the policy `POLICY-limit-resolve-routes` limits the route resolution to only routes learned through IS-IS. If you omit the import policy, all routes in inet.0 are evaluated and potentially used to resolve the protocol next hop. If you do not want to resolve against all entries, you use a policy to filter for a subset of the routes from the tables that are used for route resolution.

One common example is when you resolve against all routes in inet.0, except the default route (0/0).

Although the `import` statement is used in this configuration, no routes are imported or copied. Rather, the `import` *`policy-name`* configuration limits the set of possible routes that can be considered for route resolution.

The `resolution rib bgp.l3vpn.0 resolution-ribs inet.0` configuration is useful when a BGP RR is not in the forwarding path. In other words, there are no ingress LSPs at the RR. Consider the case where RSVP is the label signaling protocol, and RSVP is configured full mesh at the edge routers. The RR needs to be able to reflect the routes. To do so, BGP is expected to perform a route resolvability check. If a Layer 3 VPN route is received but the nexthop is not in the inet.3 table, the route cannot be resolved. Because the router is not in the forwarding path, an effective workaround is to use the information in inet.0. The metric information in inet.0 is useful for choosing the best route, even though it cannot be used for forwarding.

An alternative approach is to make sure that LSPs are provisioned to the RR. This happens automatically if you configure LDP.

Configuration

IN THIS SECTION

CLI Quick Configuration

To quickly configure this example, copy the following commands, paste them into a text file, remove any line breaks, change any details necessary to match your network configuration, and then copy and paste the commands into the CLI at the `[edit]` hierarchy level.

Route Reflector

```
set routing-options resolution rib bgp.l3vpn.0 resolution-ribs inet.0
set routing-options resolution rib inet.0 import POLICY-limit-resolve-routes
set policy-options policy-statement POLICY-limit-resolve-routes term isis from protocol isis
set policy-options policy-statement POLICY-limit-resolve-routes term isis then accept
set policy-options policy-statement POLICY-limit-resolve-routes then reject
```

Procedure

Step-by-Step Procedure

The following example requires you to navigate various levels in the configuration hierarchy. For information about navigating the CLI, see *Using the CLI Editor in Configuration Mode* in the CLI User Guide.

To configure route resolution:

1. Configure bgp.l3vpn.0 to use the information in inet.0 instead of inet.3 to resolve routes.

   ```
   [edit routing-options resolution rib bgp.l3vpn.0]
   user@RR# set resolution-ribs inet.0
   ```

2. (Optional) Configure the routing policy.

   ```
   [edit policy-options policy-statement POLICY-limit-resolve-routes]
   user@RR# set term isis from protocol isis
   user@RR# set term isis then accept
   user@RR# set then reject
   ```

3. (Optional) Apply the policy.

   ```
   [edit routing-options resolution rib inet.0]
   user@RR# set import POLICY-limit-resolve-routes
   ```

4. If you are done configuring the device, commit the configuration.

   ```
   [edit]
   user@RR# commit
   ```

Results

Confirm your configuration by issuing the `show policy-options` and `show routing-options` commands.

```
user@RR# show policy-options
policy-statement POLICY-limit-resolve-routes {
```

```
    term isis {
        from protocol isis;
        then accept;
    }
    then reject;
}
```

```
user@RR# show routing-options
resolution {
    rib bgp.l3vpn.0 {
        resolution-ribs inet.0;
    }
    rib inet.0 {
        import POLICY-limit-resolve-routes;
    }
}
```

Verification

Confirm that the configuration is working properly by running the following commands:

- *show route*
- *show route forwarding-table*
- show route resolution

SEE ALSO

Example: Configuring BGP Route Reflectors

Example: Configuring Route Resolution on PE Routers | 321

Limiting the Number of Paths and Prefixes Accepted from CE Routers in Layer 3 VPNs

You can configure a maximum limit on the number of prefixes and paths that can be installed into the routing tables. Using prefix and path limits, you can curtail the number of prefixes and paths received from a CE router in a VPN. Prefix and path limits apply only to dynamic routing protocols, and are not applicable to static or interface routes.

To limit the number of paths accepted by a PE router from a CE router, include the `maximum-paths` statement:

```
maximum-paths path-limit <log-interval interval | log-only | threshold percentage>;
```

For a list of hierarchy levels at which you can configure this statement, see the statement summary section for this statement.

Specify the **log-only** option to generate warning messages only (an advisory limit). Specify the **threshold** option to generate warnings before the limit is reached. Specify the **log-interval** option to configure the minimum time interval between log messages.

There are two modes for route limits: advisory and mandatory. An advisory limit triggers warnings. A mandatory limit rejects additional routes after the limit is reached.

NOTE: Application of a route limit may result in unpredictable dynamic routing protocol behavior. For example, when the limit is reached and routes are rejected, BGP may not reinstall the rejected routes after the number of routes drops back below the limit. BGP sessions may need to be cleared.

To limit the number of prefixes accepted by a PE router from a CE router, include the `maximum-prefixes` statement:

```
maximum-prefixes prefix-limit <log-interval interval | log-only | threshold percentage>;
```

For a list of hierarchy levels at which you can configure this statement, see the statement summary section for this statement.

There are two modes for route limits: advisory and mandatory. An advisory limit triggers warnings. A mandatory limit rejects additional routes after the limit is reached.

NOTE: Application of a route limit may result in unpredictable dynamic routing protocol behavior. For example, when the limit is reached and routes are rejected, BGP may not reinstall the rejected routes after the number of routes drops back below the limit. BGP sessions may need to be cleared.

A mandatory path or prefix limit, in addition to triggering a warning message, rejects any additional paths or prefixes once the limit is reached.

NOTE: Setting a path or prefix limit might result in unpredictable dynamic routing protocol behavior.

You can also configure the following options for both the **maximum-paths** and `maximum-prefixes` statements:

- **log-interval**—Specify the interval at which log messages are sent. This option generates warning messages only (an advisory limit).

  Specify the **log-interval** option to configure the minimum time interval between log messages.

- **log-only**—Generate warning messages only. No limit is placed on the number of paths or prefixes stored in the routing tables.
- **threshold**—Generate warning messages after the specified percentage of the maximum paths or prefixes has been reached.

Enabling Internet Access for Layer 3 VPNs

IN THIS SECTION

This topic provides examples on configuring a provider edge (PE) router to provide Internet access to customer edge (CE) routers in a VPN and configuring a router to route internet traffic to CE routers through a network address translator (NAT). The method you use depends on the needs and specifications of the individual network.

Non-VRF Internet Access Through Layer 3 VPNs

IN THIS SECTION

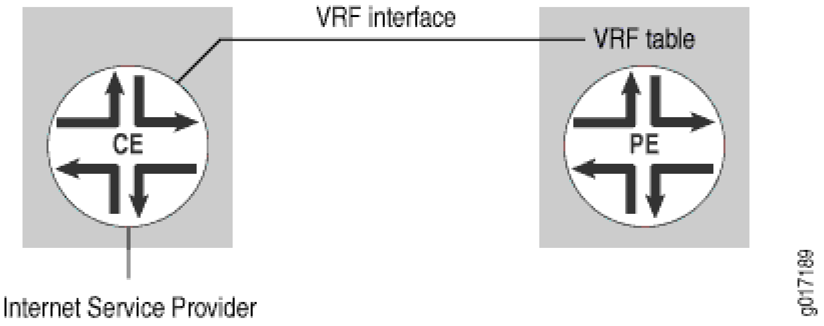
Junos OS supports Internet access from a Layer 3 virtual private network (VPN). You also need to configure the `next-table` statement at the `[edit routing-instances` *`routing-instance-name`* `routing-options static route]` hierarchy level. When configured, this statement can point a default route from the VPN table (routing instance) to the main routing table (default instance) inet.0. The main routing table stores all Internet routes and is where final route resolution occurs.

The following sections describe ways to provide Internet access to a CE router in a Layer 3 VPN without using the VPN routing and forwarding (VRF) interface. Because these methods effectively bypass the Layer 3 VPN, they are not discussed in detail.

CE Router Accesses Internet Independently of the PE Router

In this configuration, the PE router does not provide the Internet access. The CE router sends Internet traffic either to another service provider, or to the same service provider but a different router. The PE router handles Layer 3 VPN traffic only (see Figure 30 on page 331).

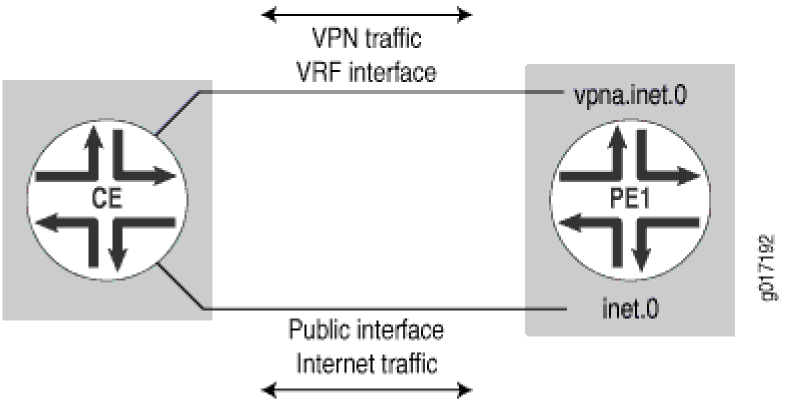
Figure 30: PE Router Does Not Provide Internet Access



PE Router Provides Layer 2 Internet Service

In this configuration, the PE router acts as a Layer 2 device, providing a Layer 2 connection (such as circuit cross-connect [CCC]) to another router that has a full set of Internet routes. The CE router can use just one physical interface and two logical interfaces to the PE router, or it can use multiple physical interfaces to the PE router (see Figure 31 on page 332).

Figure 31: PE Router Connects to a Router Connected to the Internet



Distributed Internet Access Through Layer 3 VPNs

In this scenario, the PE routers provide Internet access to the CE routers. In the examples that follow, it is assumed that the Internet routes (or defaults) are present in the inet.0 table of the PE routers that provide Internet access to selected CE routers.

When accessing the Internet from a VPN, Network Address Translation (NAT) must be performed between the VPN’s private addresses and the public addresses used on the Internet unless the VPN is using the public address space. This section includes several examples of how to provide Internet access for VPNs, most of which require that the CE routers perform the address translation. The "Routing Internet Traffic Through a Separate NAT Device" on page 354 example, however, requires that the service provider supply the NAT functionality using a NAT device connected to the PE router.

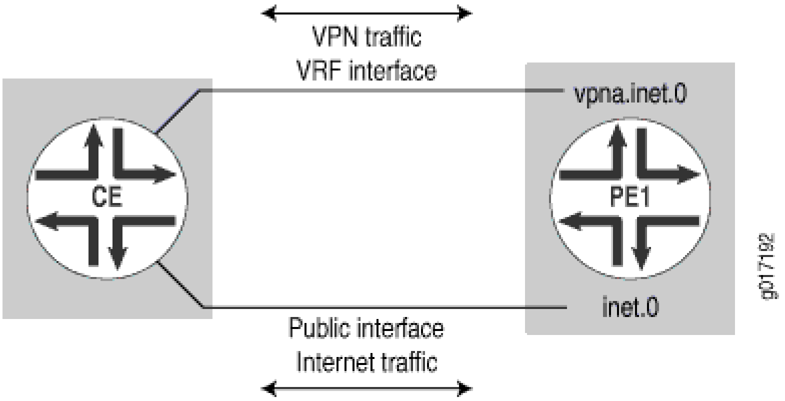
In all of the examples, the VPN’s public IP address pool (whose entries correspond to the translated private addresses) must be added to the inet.0 table and propagated to the Internet routers to receive reverse traffic from public destinations.

Routing VPN and Internet Traffic Through Different Interfaces for Layer 3 VPNs

IN THIS SECTION

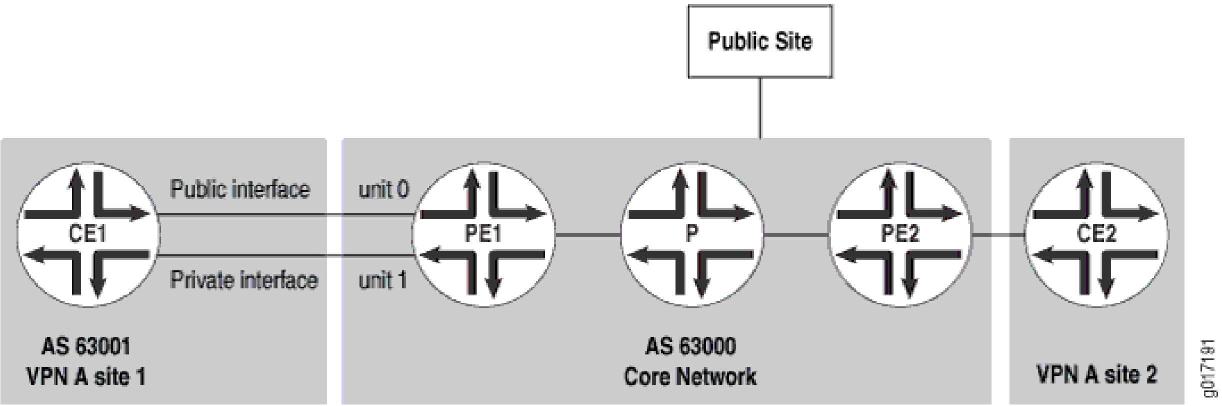
In this example, VPN and Internet traffic are routed through different interfaces. The CE router sends the VPN traffic through the VPN interface and sends the Internet traffic through a separate interface that is part of the main routing table on Router PE1 (the CE router can use either one physical interface with two logical units or two physical interfaces). NAT also occurs on the CE router (see Figure 32 on page 333).

Figure 32: Routing VPN and Internet Traffic Through Different Interfaces



The PE router is configured to install and advertise the public IP address pool for the VPN to other core routers (for return traffic). The VPN traffic is routed normally. Figure 33 on page 334 illustrates the PE router’s VPN configuration.

Figure 33: Example of Internet Traffic Routed Through Separate Interfaces



The configuration in this example has the following features:

- Router PE1 uses two logical interfaces to connect to Router CE1 using Frame Relay encapsulation.
- The routing protocol between Router PE1 and Router CE1 is EBGP.
- Router CE1's public IP address pool is `10.12.1.1` through `10.12.1.254` (`10.12.1.0/24`).
- The `next-hop-self` setting is derived from the `fix-nh policy` statement on Router PE1. PE routers are forced to use `next-hop-self` so that next-hop resolution is done only for the PE router's loopback address for non-VPN routes (by default, VPN–Internet Protocol version 4 [IPv4] routes are sent by means of `next-hop-self`).

You can configure Router CE1 with a static default route pointing to its public interface for everything else.

The following sections show how to route VPN and Internet traffic through different interfaces:

Configuring Interfaces on Router PE1

Configure an interface to handle VPN traffic and an interface to handle Internet traffic:

```
[edit]
interfaces {
    t3-0/2/0 {
        dce;
        encapsulation frame-relay;
        unit 0 {
            description "to CE1 VPN interface";
            dlci 10;
            family inet {
```

```
                address 192.168.197.13/30;
            }
        }
        unit 1 {
            description "to CE1 public interface";
            dlci 20;
            family inet {
                address 192.168.198.201/30;
            }
        }
    }
}
```

Configuring Routing Options on Router PE1

Configure a static route on Router PE1 to install a route to the CE router’s public IP address pool in inet.0:

```
[edit]
routing-options {
    static {
        route 10.12.1.0/24 next-hop 192.168.198.202;
    }
}
```

Configuring BGP, IS-IS, and LDP Protocols on Router PE1

Configure BGP on Router PE1 to allow non-VPN and VPN peering and to advertise the VPN’s public IP address pool:

```
[edit]
protocols {
    bgp {
        group pe-pe {
            type internal;
            local-address 10.255.14.171;
            family inet {
                any;
            }
            family inet-vpn {
```

```
                any;
            }
            export [fix-nh redist-static];
            neighbor 10.255.14.177;
            neighbor 10.255.14.179;
        }
    }
}
```

Configure IS-IS on Router PE1 to allow access to internal routes:

```
[edit protocols]
isis {
    level 1 disable;
    interface so-0/0/0.0;
    interface lo0.0;
}
```

Configure LDP on Router PE1 to tunnel VPN routes:

```
[edit protocols]
ldp {
    interface so-0/0/0.0;
}
```

Configuring a Routing Instance on Router PE1

Configure a routing instance on Router PE1:

```
[edit]
routing-instances {
    vpna {
        instance-type vrf;
        interface t3-0/2/0.0;
        route-distinguisher 10.255.14.171:100;
        vrf-import vpna-import;
        vrf-export vpna-export;
        protocols {
            bgp {
                group to-CE1 {
```

```
                    peer-as 63001;
                    neighbor 192.168.197.14;
                }
            }
        }
    }
}
```

Configuring Policy Options on Router PE1

You need to configure policy options on Router PE1. The `fix-nh` policy statement sets `next-hop-self` for all non-VPN routes:

```
[edit]
policy-options {
    policy-statement fix-nh {
        then {
            next-hop self;
        }
    }
}
```

The `redist-static` policy statement advertises the VPN's public IP address pool:

```
[edit policy-options]
policy-statement redist-static {
    term a {
        from {
            protocol static;
            route-filter 10.12.1.0/24 exact;
        }
        then accept;
    }
    term b {
        then reject;
    }
}
```

Configure import and export policies for `vpna`:

```
[edit policy-options]
policy-statement vpna-import {
    term a {
        from {
            protocol bgp;
            community vpna-comm;
        }
        then accept;
    }
    term b {
        then reject;
    }
}
policy-statement vpna-export {
    term a {
        from protocol bgp;
        then {
            community add vpna-comm;
            accept;
        }
    }
    term b {
        then reject;
    }
}
community vpna-comm members target:63000:100;
```

Traffic Routed by Different Interfaces: Configuration Summarized by Router

Router PE1

Interfaces

```
interfaces {
    t3-0/2/0 {
        dce;
        encapsulation frame-relay;
        unit 0 {
```

```
            description "to CE1 VPN interface";
            dlci 10;
            family inet {
                address 192.168.197.13/30;
            }
        }
        unit 1 {
            description "to CE1 public interface";
            dlci 20;
            family inet {
                address 192.168.198.201/30;
            }
        }
    }
}
```

Routing Options

```
routing-options {
    static {
        route 10.12.1.0/24 next-hop 192.168.198.202;
    }
}
```

BGP Protocol

```
protocols {
    bgp {
        group pe-pe {
            type internal;
            local-address 10.255.14.171;
            family inet {
                any;
            }
            family inet-vpn {
                any;
            }
            export [ fix-nh redist-static];
            neighbor 10.255.14.177;
            neighbor 10.255.14.179;
        }
```

```
    }
}
```

IS-IS Protocol

```
isis {
    level 1 disable;
    interface so-0/0/0.0;
    interface lo0.0;
}
```

LDP Protocol

```
ldp {
    interface so-0/0/0.0;
}
```

Routing Instance

```
routing-instances {
    vpna {
        instance-type vrf;
        interface t3-0/2/0.0;
        route-distinguisher 10.255.14.171:100;
        vrf-import vpna-import;
        vrf-export vpna-export;
        protocols {
            bgp {
                group to-CE1 {
                    peer-as 63001;
                    neighbor 192.168.197.14;
                }
            }
        }
    }
}
```

Policy Options/Policy Statements

```
policy-options {
    policy-statement fix-nh {
        then {
            next-hop self;
        }
    }
    policy-statement redist-static {
        term a {
            from {
                protocol static;
                route-filter 10.12.1.0/24 exact;
            }
            then accept;
        }
        term b {
            then reject;
        }
    }
}
```

Import and Export Policies

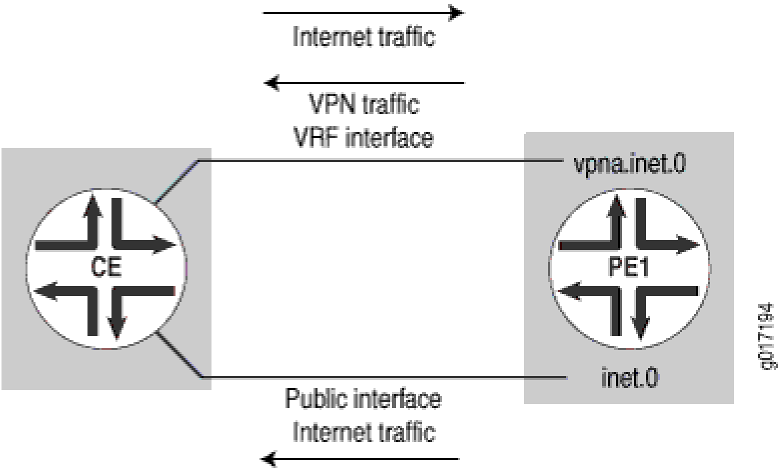
```
policy-statement vpna-import {
    term a {
        from {
            protocol bgp;
            community vpna-comm;
        }
        then accept;
    }
    term b {
        then reject;
    }
}
policy-statement vpna-export {
    term a {
        from protocol bgp;
        then {
            community add vpna-comm;
```

```
            accept;
        }
    }
    term b {
        then reject;
    }
}
community vpna-comm members target:63000:100;
```

Routing VPN and Outgoing Internet Traffic Through the Same Interface and Routing Return Internet Traffic Through a Different Interface

In this example, the CE router sends VPN and Internet traffic through the same interface but receives return Internet traffic through a different interface. The PE router has a default route in the VRF table pointing to the main routing table inet.0. It routes the VPN public IP address pool (return Internet traffic) through a different interface in inet.0 (see Figure 34 on page 342). The CE router still performs NAT functions.

Figure 34: VPN and Outgoing Internet Traffic Routed Through the Same Interface and Return Internet Traffic Routed Through a Different Interface



The following section shows how to route VPN and outgoing Internet traffic through the same interface and routing return Internet traffic through a different interface:

Configuration for Router PE1

This example has the same configuration as Router PE1 in "Routing VPN and Internet Traffic Through Different Interfaces for Layer 3 VPNs" on page 333. It uses the topology shown in "Routing VPN and Internet Traffic Through Different Interfaces for Layer 3 VPNs" on page 333. The default route to the VPN routing table is configured differently. At the `[edit routing-instances` *`routing-instance-name`* `routing-options]` hierarchy level, you configure a default static route that is installed in vpna.inet.0 and points to inet.0 for resolution:

```
[edit]
routing-instances {
    vpna {
        instance-type vrf;
        interface t3-0/2/0.0;
        route-distinguisher 10.255.14.171:100;
        vrf-import vpna-import;
        vrf-export vpna-export;
        routing-options {
            static {
                route 0.0.0.0/0 next-table inet.0;
            }
        }
        protocols {
            bgp {
                group to-CE1 {
                    peer-as 63001;
                    neighbor 192.168.197.14;
                }
            }
        }
    }
}
```

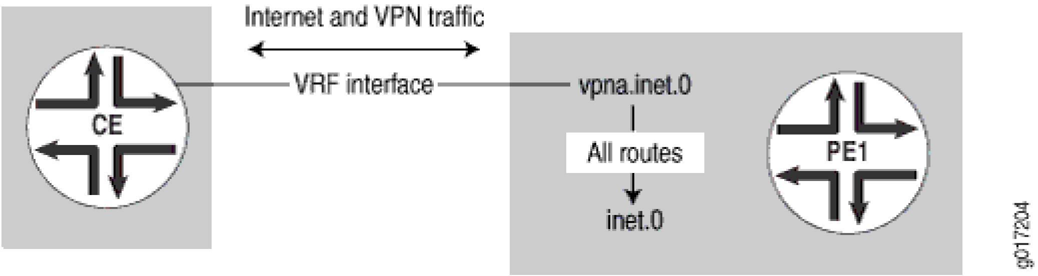
You also need to change the configuration of Router CE1 (from the configuration that works with the configuration for Router PE1 described in "Routing VPN and Internet Traffic Through Different Interfaces for Layer 3 VPNs" on page 333) to account for the differences in the configuration of the PE routers.

Routing VPN and Internet Traffic Through the Same Interface Bidirectionally (VPN Has Public Addresses)

IN THIS SECTION

This section shows how to configure a single logical interface to handle VPN and Internet traffic traveling both to and from the Internet and the CE router. This interface can handle both VPN and Internet traffic as long as there are no private addresses in the VPN. The VPN routes received from the CE router are added to the main routing table inet.0 by means of routing table groups. This allows the PE router to attract the return traffic from the Internet (see Figure 35 on page 344).

Figure 35: Interface Configured to Carry Both Internet and VPN Traffic



In this example, the CE router does not need to perform NAT, because all the VPN routes are public. The CE router has a single interface to the PE router, to which it advertises VPN routes. The PE router has a default route in the VRF table pointing to the main routing table inet.0. The PE router also imports VPN routes received from the CE router into inet.0 by means of routing table groups.

The following configuration for Router PE1 uses the same topology as in "Routing VPN and Internet Traffic Through Different Interfaces for Layer 3 VPNs" on page 333. This configuration uses a single logical interface (instead of two) between Router PE1 and Router CE1.

The following sections show how to route VPN and Internet traffic through the same interface bidirectionally (VPN has public addresses):

Configuring Routing Options on Router PE1

Configure a routing table group definition for installing VPN routes in routing table groups vpna.inet.0 and inet.0:

```
[edit]
routing-options {
    rib-groups {
        vpna-to-inet0 {
            import-rib [ vpna.inet.0 inet.0 ];
        }
    }
}
```

Configuring Routing Protocols on Router PE1

Configure MPLS, BGP, IS-IS, and LDP protocols on Router PE1. This configuration does not include the `policy redist-static` statement at the `[edit protocols bgp group pe-pe]` hierarchy level. The VPN routes are sent directly to IBGP.

Configure BGP on Router PE1 to allow non-VPN and VPN peering, and to advertise the VPN's public IP address pool:

```
[edit]
protocols {
    mpls {
        interface so-0/0/0.0;
    }
    bgp {
        group pe-pe {
            type internal;
            local-address 10.255.14.171;
            family inet {
                any;
            }
            family inet-vpn {
                any;
            }
            export fix-nh;
            neighbor 10.255.14.177;
            neighbor 10.255.14.173;
```

```
        }
    }
    isis {
        level 1 disable;
        interface so-0/0/0.0;
        interface lo0.0;
    }
    ldp {
        interface so-0/0/0.0;
    }
}
```

Configuring the Routing Instance on Router PE1

This section describes how to configure the routing instance on Router PE1. The static route defined in the `routing-options` statement directs Internet traffic from the CE router to the inet.0 routing table. The routing table group defined by the `rib-group vpna-to-inet0` statement adds the VPN routes to inet.0.

Configure the routing instance on Router PE1:

```
[edit]
routing-instances {
    vpna {
        instance-type vrf;
        interface t3-0/2/0.0;
        route-distinguisher 10.255.14.171:100;
        vrf-import vpna-import;
        vrf-export vpna-export;
        routing-options {
            static {
                route 0.0.0.0/0 next-table inet.0;
            }
        }
        protocols {
            bgp {
                group to-CE1 {
                    family inet {
                        unicast {
                            rib-group vpna-to-inet0;
                        }
                    }
                    peer-as 63001;
```

```
                    neighbor 192.168.197.14;
                }
            }
        }
    }
}
```

You must configure Router CE1 to forward all traffic to Router PE1 using a default route. Alternatively, the default route can be advertised from Router PE1 to Router CE1 with EBGP.

Traffic Routed Through the Same Interface Bidirectionally: Configuration Summarized by Router

Router PE1

This example uses the same configuration as in "Routing VPN and Internet Traffic Through Different Interfaces for Layer 3 VPNs" on page 333. This configuration uses a single logical interface (instead of two) between Router PE1 and Router CE1.

Routing Options

```
routing-options {
    rib-groups {
        vpna-to-inet0 {
            import-rib [ vpna.inet.0 inet.0 ];
        }
    }
}
```

Routing Protocols

```
protocols {
    mpls {
        interface so-0/0/0.0;
    }
    bgp {
        group pe-pe {
            type internal;
            local-address 10.255.14.171;
            family inet {
                any;
            }
```

```
            family inet-vpn {
                any;
            }
            export fix-nh;
            neighbor 10.255.14.177;
            neighbor 10.255.14.173;
        }
    }
    isis {
        level 1 disable;
        interface so-0/0/0.0;
        interface lo0.0;
    }
    ldp {
        interface so-0/0/0.0;
    }
}
```

Routing Instance

```
routing-instances {
    vpna {
        instance-type vrf;
        interface t3-0/2/0.0;
        route-distinguisher 10.255.14.171:100;
        vrf-import vpna-import;
        vrf-export vpna-export;
        routing-options {
            static {
                route 0.0.0.0/0 next-table inet.0;
            }
        }
        protocols {
            bgp {
                group to-CE1 {
                    family inet {
                        unicast {
                            rib-group vpna-to-inet0;
                        }
                    }
                    peer-as 63001;
                    neighbor 192.168.197.14;
```

```
                    }
                }
            }
        }
    }
```

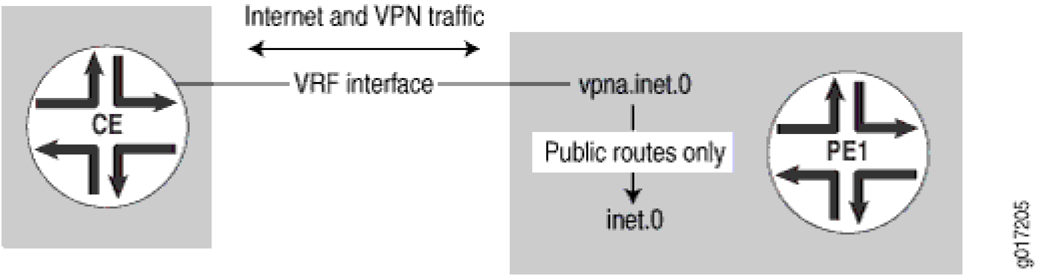
Routing VPN and Internet Traffic Through the Same Interface Bidirectionally (VPN Has Private Addresses)

IN THIS SECTION

The example in this section shows how to route VPN and Internet traffic through the same interface in both directions (from the CE router to the Internet and from the Internet to the CE router). The VPN in this example has private addresses. If you can configure EBGP on the CE router, you can configure a PE router using the configuration outlined in "Routing VPN and Internet Traffic Through the Same Interface Bidirectionally (VPN Has Public Addresses)" on page 344, even if the VPN has private addresses.

In the example described in this section, the CE router uses separate communities to advertise its VPN routes and public routes. The PE router selectively imports only the public routes into the inet.0 routing table. This configuration ensures that return traffic from the Internet uses the same interface between the PE and CE routers as that used by VPN traffic going out to public Internet addresses (see Figure 36 on page 350).

Figure 36: VPN and Internet Traffic Routed Through the Same Interface



In this example, the CE router has one interface and a BGP session with the PE router, and it tags VPN routes and Internet routes with different communities. The PE router has one interface, selectively imports routes for the VPN's public IP address pool into inet.0, and has a default route in the VRF routing table pointing to inet.0.

The following sections show how to route VPN and Internet traffic through the same interface bidirectionally (VPN has private addresses):

Configuring Routing Options for Router PE1

On Router PE1, configure a routing table group to install VPN routes in the vpna.inet.0 and inet.0 routing tables:

```
[edit]
routing-options {
    rib-groups {
        vpna-to-inet0 {
            import-policy import-public-addr-to-inet0;
            import-rib [ vpna.inet.0 inet.0 ];
        }
    }
}
```

Configuring a Routing Instance for Router PE1

On Router PE1, configure a routing instance. As part of the configuration for the routing instance, configure a static route that is installed in vpna.inet.0 and is pointed at inet.0 for resolution.

```
[edit]
routing-instances {
    vpna {
```

```
        instance-type vrf;
        interface t3-0/2/0.0;
        route-distinguisher 10.255.14.171:100;
        vrf-import vpna-import;
        vrf-export vpna-export;
        routing-options {
            static {
                route 0.0.0.0/0 next-table inet.0;
            }
        }
    }
}
```

At the `[edit routing-instances vpna protocols bgp]` hierarchy level, configure a policy (`import-public-addr-to-inet0`) to import public routes into inet.0 and a routing table group (`vpna-to-inet0`) to allow BGP to install routes into multiple routing tables (vpna.inet.0 and inet.0):

```
[edit routing-instances vpna]
protocols {
    bgp {
        group to-CE1 {
            import import-public-addr-to-inet0;
            family inet {
                unicast {
                    rib-group vpna-to-inet0;
                }
            }
            peer-as 63001;
            neighbor 192.168.197.14;
        }
    }
}
```

Configuring Policy Options for Router PE1

Configure the policy options for Router PE1 to accept all routes initially (`term a`) and then to install routes with a `public-comm` community into routing table inet.0 (`term b`):

```
[edit]
policy-options {
```

```
    policy-statement import-public-addr-to-inet0 {
        term a {
            from {
                protocol bgp;
                rib vpna.inet.0;
                community [ public-comm private-comm ];
            }
            then accept;
        }
        term b {
            from {
                protocol bgp;
                community public-comm;
            }
            to rib inet.0;
            then accept;
        }
        term c {
            then reject;
        }
    }
    community private-comm members target:1:333;
    community public-comm members target:1:111;
    community vpna-comm members target:63000:100;
}
```

Traffic Routed by the Same Interface Bidirectionally (VPN Has Private Addresses): Configuration Summarized by Router

Router PE1

Routing Options

```
[edit]
routing-options {
    rib-groups {
        vpna-to-inet0 {
            import-policy import-public-addr-to-inet0;
            import-rib [ vpna.inet.0 inet.0 ];
        }
```

```
    }
}
```

Routing Instances

```
[edit]
routing-instances {
    vpna {
        instance-type vrf;
        interface t3-0/2/0.0;
        route-distinguisher 10.255.14.171:100;
        vrf-import vpna-import;
        vrf-export vpna-export;
        routing-options {
            static {
                route 0.0.0.0/0 next-table inet.0;
            }
        }
    }
}
```

Routing Instances Protocols BGP

```
[edit routing-instances vpna]
protocols {
    bgp {
        group to-CE1 {
            family inet {
                unicast {
                    rib-group vpna-to-inet0;
                }
            }
            peer-as 63001;
            neighbor 192.168.197.14;
        }
    }
}
```

Policy Options

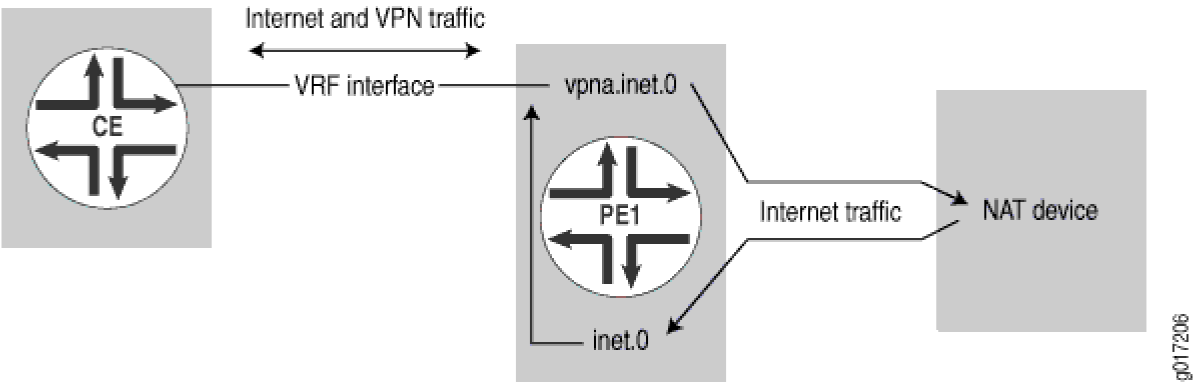
```
[edit]
policy-options {
    policy-statement import-public-addr-to-inet0 {
        term a {
            from {
                protocol bgp;
                rib vpna.inet.0;
                community [ public-comm private-comm ];
            }
            then accept;
        }
        term b {
            from {
                protocol bgp;
                community public-comm;
            }
            to rib inet.0;
            then accept;
        }
        term c {
            then reject;
        }
    }
    community private-comm members target:1:333;
    community public-comm members target:1:111;
    community vpna-comm members target:63000:100;
}
```

Routing Internet Traffic Through a Separate NAT Device

IN THIS SECTION

In this example, the CE router does not perform NAT. It sends both VPN and Internet traffic over the same interface to the PE router. The PE router is connected to an NAT device by means of two interfaces. One interface is configured in the PE router's VRF table and points to a VPN interface on the NAT device, which can route Internet traffic for the VPN. The other interface is in a default instance; for example, part of public routing table inet.0. There can be a single physical connection between the PE router and the NAT device and multiple logical connections—one for each VRF table and another interface—as part of the global routing table (see Figure 37 on page 355).

Figure 37: Internet Traffic Routed Through a Separate NAT Device



Requirements

This example uses the following hardware and software components:

- M Series routers
- Junos OS Release 9.3 or later

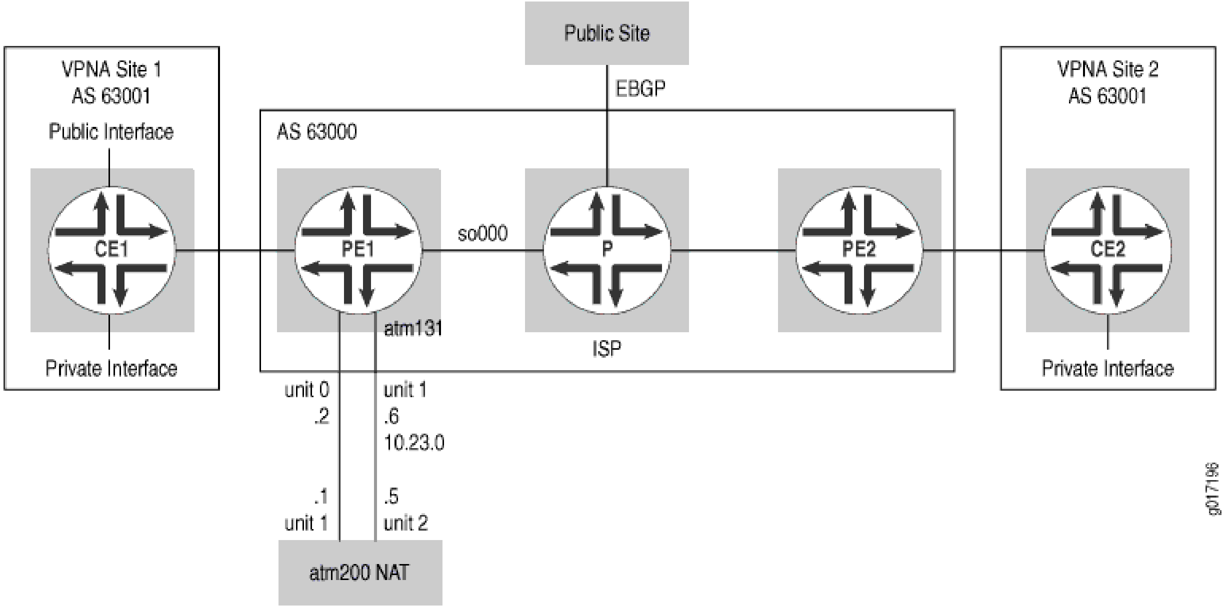
Overview

IN THIS SECTION

This example's topology expands upon that illustrated in "Routing VPN and Internet Traffic Through Different Interfaces for Layer 3 VPNs" on page 333. The CE router sends both VPN and Internet traffic to Router PE1. VPN traffic is routed based on the VPN routes received by Router PE1. Traffic for everything else is sent to the NAT device using Router PE1's private interface to the NAT device, which then translates the private addresses and sends the traffic back to Router PE1 using that router's public interface (see Figure 38 on page 356).

Topology

Figure 38: Internet Traffic Routed Through a NAT Example Topology



Configuration

IN THIS SECTION

To route Internet traffic through a separate NAT device, perform these tasks:

Configuring Interfaces on Router PE1

Step-by-Step Procedure

1. Configure an interface for VPN traffic from Router CE1:

```
[edit]
interfaces {
    t3-0/2/0 {
        dce;
        encapsulation frame-relay;
        unit 0 {
            description "to CE1 VPN interface";
            dlci 10;
            family inet {
                address 192.168.197.13/30;
            }
        }
    }
}
```

2. Configure an interface for VPN traffic to and from the NAT device (unit 0), and an interface for Internet traffic to and from the NAT device (unit 1):

```
[edit]
interfaces {
    at-1/3/1 {
        atm-options {
            vpi 1 maximum-vcs 255;
        }
        unit 0 {
            description "to NAT VPN interface";
            vci 1.100;
            family inet {
                address 10.23.0.2/32 {
                    destination 10.23.0.1;
                }
            }
        }
        unit 1 {
            description "to NAT public interface";
```

```
            vci 1.101;
            family inet {
                address 10.23.0.6/32 {
                    destination 10.23.0.5;
                }
            }
        }
    }
}
```

Configuring Routing Options for Router PE1

Step-by-Step Procedure

1. Configure a static route on Router PE1 to direct Internet traffic to the CE router through the NAT device. Router PE1 distributes this route to the Internet.

```
[edit]
routing-options {
    static {
        route 10.12.1.0/24 next-hop 10.23.0.5;
    }
}
```

Configuring Routing Protocols on Router PE1

Step-by-Step Procedure

Configure the following routing protocols on Router PE1:

1. Configure MPLS on Router PE1. Include the NAT device's VPN interface in the VRF table.

```
[edit]
protocols {
    mpls {
        interface so-0/0/0.0;
        interface at-1/3/1.0;
    }
}
```

2. Configure BGP on Router PE1. Include a policy to advertise the public IP address pool:

```
[edit]
protocols {
    bgp {
        group pe-pe {
            type internal;
            local-address 10.255.14.171;
            family inet {
                any;
            }
            family inet-vpn {
                any;
            }
            export [ fix-nh redist-static ];
            neighbor 10.255.14.177;
            neighbor 10.255.14.173;
        }
    }
}
```

3. Configure IS-IS on Router PE1:

```
[edit]
protocols {
    isis {
        level 1 disable;
        interface so-0/0/0.0;
        interface lo0.0;
    }
}
```

4. Configure LDP on Router PE1:

```
[edit]
protocols {
    ldp {
        interface so-0/0/0.0;
```

```
    }
}
```

Configuring a Routing Instance on Router PE1

Step-by-Step Procedure

Configure the Layer 3 VPN routing instance on Router PE1:

1. Configure a routing instance on Router PE1. As part of the routing instance configuration, under `routing-options`, configure a static default route in vpna.inet.0 pointing to the NAT device's VPN interface (this directs all non-VPN traffic to the NAT device):

```
[edit]
routing-instances {
    vpna {
        instance-type vrf;
        interface t3-0/2/0.0;
        interface at-1/3/1.0;
        route-distinguisher 10.255.14.171:100;
        vrf-import vpna-import;
        vrf-export vpna-export;
        routing-options {
            static {
                route 0.0.0.0/0 next-hop 10.23.0.1;
            }
        }
        protocols {
            bgp {
                group to-CE1 {
                    peer-as 63001;
                    neighbor 192.168.197.14;
                }
            }
        }
    }
}
```

2. Configure the routing policy for the Layer 3 VPN routing instance on Router PE1:

```
policy-options {
    policy-statement fix-nh {
        then {
            next-hop self;
        }
    }
    policy-statement redist-static {
        term a {
            from {
                protocol static;
                route-filter 10.12.1.0/24 exact;
            }
            then accept;
        }
        term b {
            from protocol bgp;
            then accept;
        }
        term c {
            then accept;
        }
    }
    policy-statement vpna-import {
        term a {
            from {
                protocol bgp;
                community vpna-comm;
            }
            then accept;
        }
        term b {
            then reject;
        }
    }
    policy-statement vpna-export {
        term a {
            from protocol bgp;
            then {
                community add vpna-comm;
                accept;
```

```
            }
        }
        term b {
            then reject;
        }
    }
    community vpna-comm members target:63000:100;
}
```

Results

From configuration mode on Router PE1, confirm your configuration by entering the show interfaces, show routing-options, show protocols, show routing-instances and show policy-options commands. If the output does not display the intended configuration, repeat the instructions in this example to correct the configuration.

```
user@PE1# show interfaces
interfaces {
    t3-0/2/0 {
        dce;
        encapsulation frame-relay;
        unit 0 {
            description "to CE1 VPN interface";
            dlci 10;
            family inet {
                address 192.168.197.13/30;
            }
        }
    }
    at-1/3/1 {
        atm-options {
            vpi 1 maximum-vcs 255;
        }
        unit 0 {
            description "to NAT VPN interface";
            vci 1.100;
            family inet {
                address 10.23.0.2/32 {
                    destination 10.23.0.1;
                }
            }
```

```
        }
        unit 1 {
            description "to NAT public interface";
            vci 1.101;
            family inet {
                address 10.23.0.6/32 {
                    destination 10.23.0.5;
                }
            }
        }
    }
}
```

```
user@PE1# show routing-options
routing-options {
    static {
        route 10.12.1.0/24 next-hop 10.23.0.5;
    }
}
```

```
user@PE1# show protocols
protocols {
    mpls {
        interface so-0/0/0.0;
        interface at-1/3/1.0;
    }
    bgp {
        group pe-pe {
            type internal;
            local-address 10.255.14.171;
            family inet {
                any;
            }
            family inet-vpn {
                any;
            }
            export [ fix-nh redist-static ];
            neighbor 10.255.14.177;
            neighbor 10.255.14.173;
        }
```

```
    }
    isis {
        level 1 disable;
        interface so-0/0/0.0;
        interface lo0.0;
    }
    ldp {
        interface so-0/0/0.0;
    }
}
```

```
user@PE1# show routing-instances
routing-instances {
    vpna {
        instance-type vrf;
        interface t3-0/2/0.0;
        interface at-1/3/1.0;
        route-distinguisher 10.255.14.171:100;
        vrf-import vpna-import;
        vrf-export vpna-export;
        routing-options {
            static {
                route 0.0.0.0/0 next-hop 10.23.0.1;
            }
        }
        protocols {
            bgp {
                group to-CE1 {
                    peer-as 63001;
                    neighbor 192.168.197.14;
                }
            }
        }
    }
}
```

```
user@PE1# show policy-options
policy-options {
    policy-statement fix-nh {
        then {
```

```
            next-hop self;
        }
    }
    policy-statement redist-static {
        term a {
            from {
                protocol static;
                route-filter 10.12.1.0/24 exact;
            }
            then accept;
        }
        term b {
            from protocol bgp;
            then accept;
        }
        term c {
            then accept;
        }
    }
    policy-statement vpna-import {
        term a {
            from {
                protocol bgp;
                community vpna-comm;
            }
            then accept;
        }
        term b {
            then reject;
        }
    }
    policy-statement vpna-export {
        term a {
            from protocol bgp;
            then {
                community add vpna-comm;
                accept;
            }
        }
        term b {
            then reject;
        }
    }
```

```
    community vpna-comm members target:63000:100;
}
```

Centralized Internet Access Through Layer 3 VPNs

IN THIS SECTION

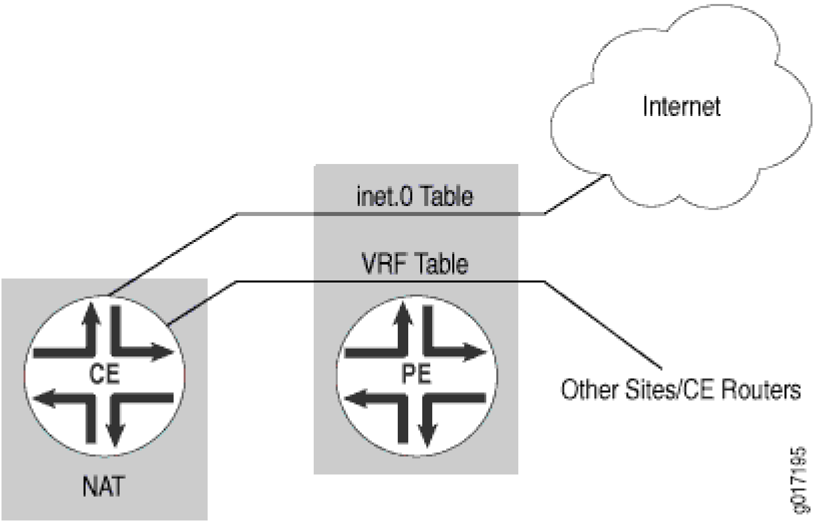
This section describes several ways to configure a CE router to act as a central site for Internet access. Internet traffic from other sites (CE routers) is routed to the hub CE router (which also performs NAT) using that router's VPN interface. The hub CE router then forwards the traffic to a PE router connected to the Internet through another interface identified in the inet.0 table. The hub CE router can advertise a default route to the spoke CE routers. The disadvantage of this type of configuration is that all traffic has to go through the central CE router before going to the Internet, causing network delays if this router receives too much traffic. However, in a corporate network, traffic might have to be routed to a central site because most corporate networks separate the VPN from the Internet by means of a single firewall.

This section includes the following examples:

Routing Internet Traffic Through a Hub CE Router

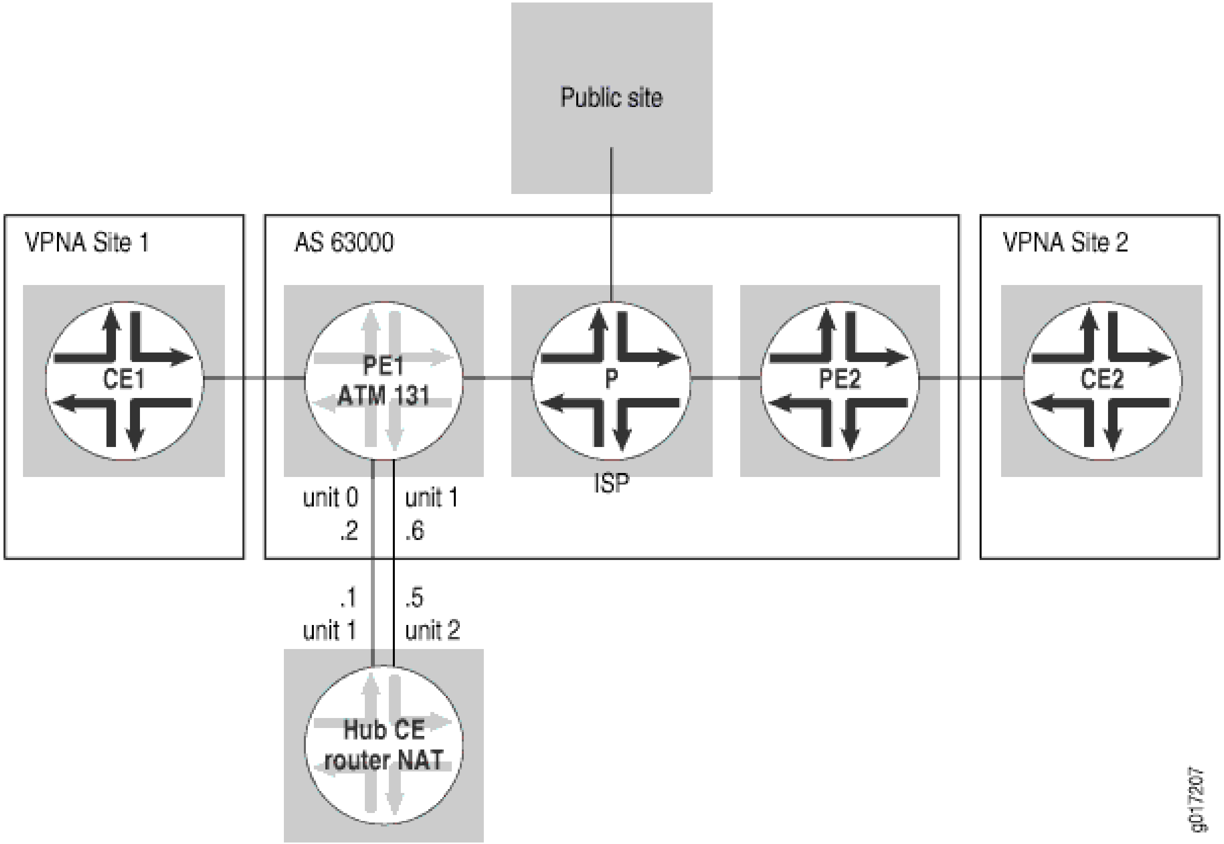
In this example, Internet traffic is routed through a hub CE router. The hub CE router has two interfaces to the hub PE router: a VPN interface and a public interface. It performs NAT on traffic forwarded from the hub PE router through the VPN interface and forwards that traffic from its public interface back to the hub PE router. The hub PE router has a static default route in its VRF table pointing to the hub CE router's VPN interface. It announces this default route to the rest of the VPN, attracting all non-VPN traffic to the hub CE route. The hub PE router also installs and distributes the VPN's public IP address space (see Figure 39 on page 367).

Figure 39: Internet Access Through a Hub CE Router Performing NAT



The configuration for this example is almost identical to that described in "Routing Internet Traffic Through a Separate NAT Device" on page 354. The difference is that Router PE1 is configured to announce a static default route to the other CE routers (see Figure 40 on page 368).

Figure 40: Internet Access Provided Through a Hub CE Router



The following sections show how to configure centralized Internet access by routing Internet traffic through a hub CE router:

Configuring a Routing Instance on Router PE1

Configure a routing instance for Router PE1. As part of this configuration, under `routing-options`, configure a default static route (`route 0.0.0.0/0`) to be installed in vpna.inet.0, and point the route to the hub CE router's VPN interface (`10.23.0.1`). Also, configure BGP under the routing instance to export the default route to the local CE router:

```
[edit]
routing-instances {
    vpna {
        instance-type vrf;
        interface t3-0/2/0.0;
        interface at-1/3/1.0;
        route-distinguisher 10.255.14.171:100;
        vrf-import vpna-import;
```

```
        vrf-export vpna-export;
        routing-options {
            static {
                route 0.0.0.0/0 next-hop 10.23.0.1;
            }
        }
        protocols {
            bgp {
                group to-CE1 {
                    export export-default;
                    peer-as 63001;
                    neighbor 192.168.197.14;
                }
            }
        }
    }
}
```

Configuring Policy Options on Router PE1

Configure policy options on Router PE1. As part of this configuration, Router PE1 should export the static default route to all the remote PE routers in `vpna` (configured in the `policy-statement vpna-export` statement under `term b`):

```
[edit]
policy-options {
    policy-statement vpna-export {
        term a {
            from protocol bgp;
            then {
                community add vpna-comm;
                accept;
            }
        }
        term b {
            from {
                protocol static;
                route-filter 0.0.0.0/0 exact;
            }
            then {
                community add vpna-comm;
```

```
                accept;
            }
        }
        term c {
            then reject;
        }
    }
    policy-statement export-default {
        term a {
            from {
                protocol static;
                route-filter 0.0.0.0/0 exact;
            }
            then accept;
        }
        term b {
            from protocol bgp;
            then accept;
        }
        term c {
            then reject;
        }
    }
}
```

Internet Traffic Routed by a Hub CE Router: Configuration Summarized by Router

Router PE1

The configuration for Router PE1 is almost identical to that for the example in "Routing Internet Traffic Through a Separate NAT Device" on page 354. The difference is that Router PE1 is configured to announce a static default route to the other CE routers.

Routing Instance

```
routing-instances {
    vpna {
        instance-type vrf;
        interface t3-0/2/0.0;
        interface at-1/3/1.0;
        route-distinguisher 10.255.14.171:100;
```

```
        vrf-import vpna-import;
        vrf-export vpna-export;
        routing-options {
            static {
                route 0.0.0.0/0 next-hop 10.23.0.1;
            }
        }
        protocols {
            bgp {
                group to-CE1 {
                    export export-default;
                    peer-as 63001;
                    neighbor 192.168.197.14;
                }
            }
        }
    }
}
```

Policy Options

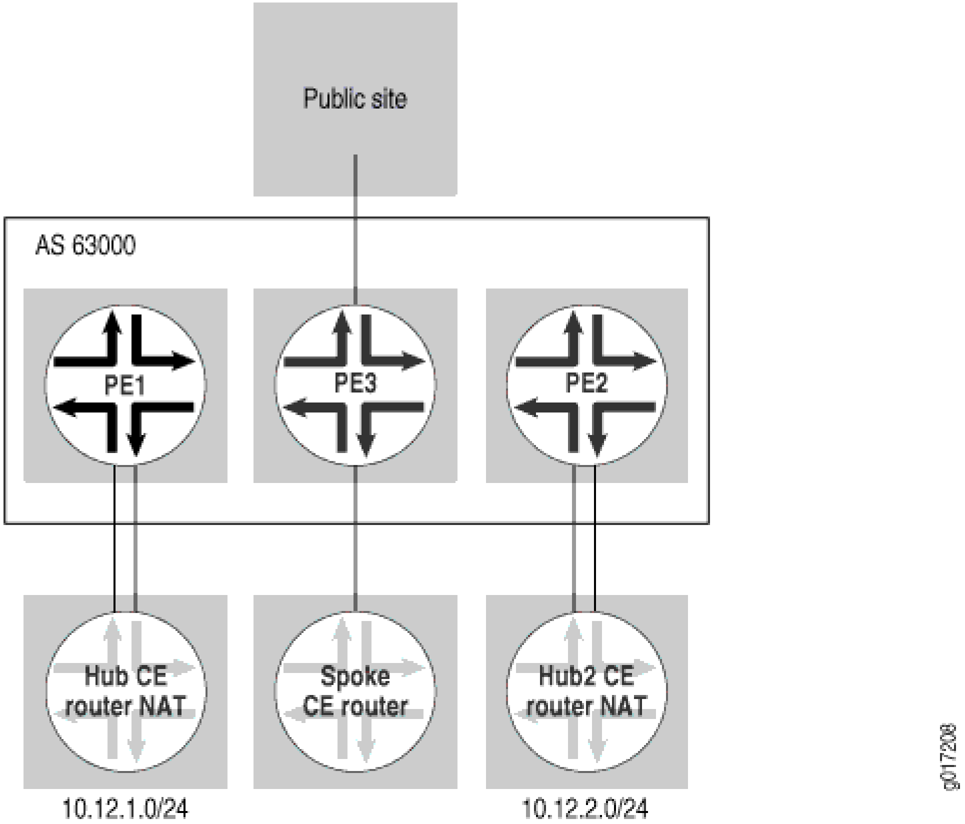
```
policy-options {
    policy-statement vpna-export {
        term a {
            from protocol bgp;
            then {
                community add vpna-comm;
                accept;
            }
        }
        term b {
            from {
                protocol static;
                route-filter 0.0.0.0/0 exact;
            }
            then {
                community add vpna-comm;
                accept;
            }
        }
        term c {
            then reject;
```

```
        }
    }
    policy-statement export-default {
        term a {
            from {
                protocol static;
                route-filter 0.0.0.0/0 exact;
            }
            then accept;
        }
        term b {
            from protocol bgp;
            then accept;
        }
        term c {
            then reject;
        }
    }
}
```

Routing Internet Traffic Through Multiple CE Routers

The example in this section is an extension of that described in "Centralized Internet Access Through Layer 3 VPNs" on page 366. This example provides different exit points for different sites by means of multiple hub CE routers that perform similar functions. Each hub CE router tags the default route with a different route target and allows the spoke CE routers to select the hub site that should be used for Internet access (see Figure 41 on page 373).

Figure 41: Two Hub CE Routers Handling Internet Traffic and NAT



This example uses two hub CE routers that handle NAT and Internet traffic:

- Hub1 CE router tags `0/0` with community `public-comm1` (target: `1:111`)
- Hub2 CE router tags `0/0` with community `public-comm2` (target: `1:112`)

The spoke CE router in this example is configured to have a bias toward Hub2 for Internet access.

The following sections describe how configure two hub CE routers to handle internet traffic and NAT:

Configuring a Routing Instance on Router PE1

Configure a routing instance on Router PE1:

```
[edit]
routing-instances {
    vpna {
        instance-type vrf;
        interface t3-0/2/0.0;
        interface at-1/3/1.0;
```

```
        route-distinguisher 10.255.14.171:100;
        vrf-import vpna-import;
        vrf-export vpna-export;
        routing-options {
            static {
                route 0.0.0.0/0 next-hop 10.23.0.1;
            }
        }
        protocols {
            bgp {
                group to-CE1 {
                    export export-default;
                    peer-as 63001;
                    neighbor 192.168.197.14;
                }
            }
        }
    }
}
```

Configuring Policy Options on Router PE1

The policy options for Router PE1 are the same as in "Routing Internet Traffic Through a Hub CE Router" on page 366, but the configuration in this example includes an additional community, `public-comm1`, in the `export` statement:

```
[edit]
policy-options {
    policy-statement vpna-import {
        term a {
            from {
                protocol bgp;
                community vpna-comm;
            }
            then accept;
        }
        term b {
            then reject;
        }
    }
    policy-statement vpna-export {
```

```
        term a {
            from {
                protocol static;
                route-filter 0.0.0.0/0 exact;
            }
            then {
                community add public-comm1;
                community add vpna-comm;
                accept;
            }
        }
        term b {
            from protocol bgp;
            then {
                community add vpna-comm;
                accept;
            }
        }
        term c {
            then reject;
        }
    }
    community public-comm1 members target:1:111;
    community public-comm2 members target:1:112;
    community vpna-comm members target:63000:100;
}
```

The configuration of Router PE2 is identical to that of Router PE1 except that Router PE2 exports the default route through community `public-comm2`.

Configuring a Routing Instance on Router PE3

Configure routing instance `vpna` on Router PE3:

```
[edit]
routing-instances {
    vpna {
        instance-type vrf;
        interface t1-0/2/0.0;
        route-distinguisher 10.255.14.173:100;
        vrf-import vpna-import;
        vrf-export vpna-export;
```

```
        protocols {
            rip {
                group to-vpn12 {
                    export export-CE;
                    neighbor t1-0/2/0.0;
                }
            }
        }
    }
}
```

Configuring Policy Options on Router PE3

Configure the `vrf-import` policy for Router PE3 to select the Internet exit point based on the additional communities specified in "Configuring Policy Options on Router PE1" on page 374:

```
[edit]
policy-options {
    policy-statement vpna-export {
        term a {
            from protocol rip;
            then {
                community add vpna-comm;
                accept;
            }
        }
        term b {
            then reject;
        }
    }
    policy-statement vpna-import {
        term a {
            from {
                protocol bgp;
                community public-comm1;
                route-filter 0.0.0.0/0 exact;
            }
            then reject;
        }
        term b {
            from {
```

```
                protocol bgp;
                community vpna-comm;
            }
            then accept;
        }
        term c {
            then reject;
        }
    }
    policy-statement export-CE {
        from protocol bgp;
        then accept;
    }
    community vpna-comm members target:69:100;
    community public-comm1 members target:1:111;
    community public-comm2 members target:1:112;
}
```

Routing Internet Traffic Through Multiple CE Routers: Configuration Summarized by Router

Router PE1

This configuration is an extension of the example in "Routing Internet Traffic Through a Hub CE Router" on page 366. It provides different exit points for various sites by using multiple hub CE routers that perform similar functions.

Routing Instances

```
routing-instances {
    vpna {
        instance-type vrf;
        interface t3-0/2/0.0;
        interface at-1/3/1.0;
        route-distinguisher 10.255.14.171:100;
        vrf-import vpna-import;
        vrf-export vpna-export;
        routing-options {
            static {
                route 0.0.0.0/0 next-hop 10.23.0.1;
            }
        }
```

```
        protocols {
            bgp {
                group to-CE1 {
                    export export-default;
                    peer-as 63001;
                    neighbor 192.168.197.14;
                }
            }
        }
    }
}
```

Policy Options

```
policy-options {
    policy-statement vpna-import {
        term a {
            from {
                protocol bgp;
                community vpna-comm;
            }
            then accept;
        }
        term b {
            then reject;
        }
    }
    policy-statement vpna-export {
        term a {
            from {
                protocol static;
                route-filter 0.0.0.0/0 exact;
            }
            then {
                community add public-comm1;
                community add vpna-comm;
                accept;
            }
        }
        term b {
            from protocol bgp;
            then {
```

```
                community add vpna-comm;
                accept;
            }
        }
        term c {
            then reject;
        }
    }
    community public-comm1 members target:1:111;
    community public-comm2 members target:1:112;
    community vpna-comm members target:63000:100;
}
```

Router PE2

The configuration of Router PE2 is identical to that of Router PE1, except that Router PE2 exports the default route through community `public-comm2`.

Router PE3

Routing Instances

```
routing-instances {
    vpna {
        instance-type vrf;
        interface t1-0/2/0.0;
        route-distinguisher 10.255.14.173:100;
        vrf-import vpna-import;
        vrf-export vpna-export;
        protocols {
            rip {
                group to-vpn12 {
                    export export-CE;
                    neighbor t1-0/2/0.0;
                }
            }
        }
    }
}
```

Policy Options

```
policy-options {
    policy-statement vpna-export {
        term a {
            from protocol rip;
            then {
                community add vpna-comm;
                accept;
            }
        }
        term b {
            then reject;
        }
    }
    policy-statement vpna-import {
        term a {
            from {
                protocol bgp;
                community public-comm1;
                route-filter 0.0.0.0/0 exact;
            }
            then reject;
        }
        term b {
            from {
                protocol bgp;
                community vpna-comm;
            }
            then accept;
        }
        term c {
            then reject;
        }
    }
    policy-statement export-CE {
        from protocol bgp;
        then accept;
    }
    community vpna-comm members target:69:100;
    community public-comm1 members target:1:111;
```

```
    community public-comm2 members target:1:112;
}
```

Connecting Layer 3 VPNs to Layer 2 Circuits

IN THIS SECTION

Applications for Interconnecting a Layer 2 Circuit with a Layer 3 VPN

MPLS-based Layer 2 services are growing in demand among enterprise and service providers. This creates new challenges related to interoperability between Layer 2 and Layer 3 services for service providers who want to provide end-to-end value-added services. There are various reasons to stitch different Layer 2 services to one another and to Layer 3 services. For example, to expand the service offerings and to expand geographically. The Junos OS has various features to address the needs of the service provider.

You can enable pseudowire services and configure a pseduowire service interface as an access point for interconnecting layer 2 circuits to layer 3 VPNs. For more information, see Pseudowire Subscriber Logical Interfaces Overview.

Interconnecting a Layer 2 Circuit with a Layer 3 VPN provides the following benefits:

- Interconnecting a Layer 2 Circuit with a Layer 3 VPN enables the sharing of a service provider's core network infrastructure between IP and Layer 2 circuit services, reducing the cost of providing those services. A Layer 2 MPLS circuit allows service providers to create a Layer 2 circuit service over an existing IP and MPLS backbone.
- Service providers do not have to invest in separate Layer 2 equipment to provide Layer 2 circuit service. A service provider can configure a provider edge router to run any Layer 3 protocol in addition to the Layer 2 protocols. Customers who prefer to maintain control over most of the administration of their own networks want Layer 2 circuit connections with their service provider instead of a Layer 3 VPN connection.

Example: Interconnecting a Layer 2 Circuit with a Layer 3 VPN

IN THIS SECTION

This example provides a step-by-step procedure and commands for configuring and verifying a Layer 2 circuit to Layer 3 VPN interconnection. It contains the following sections:

Requirements

This example uses the following hardware and software components:

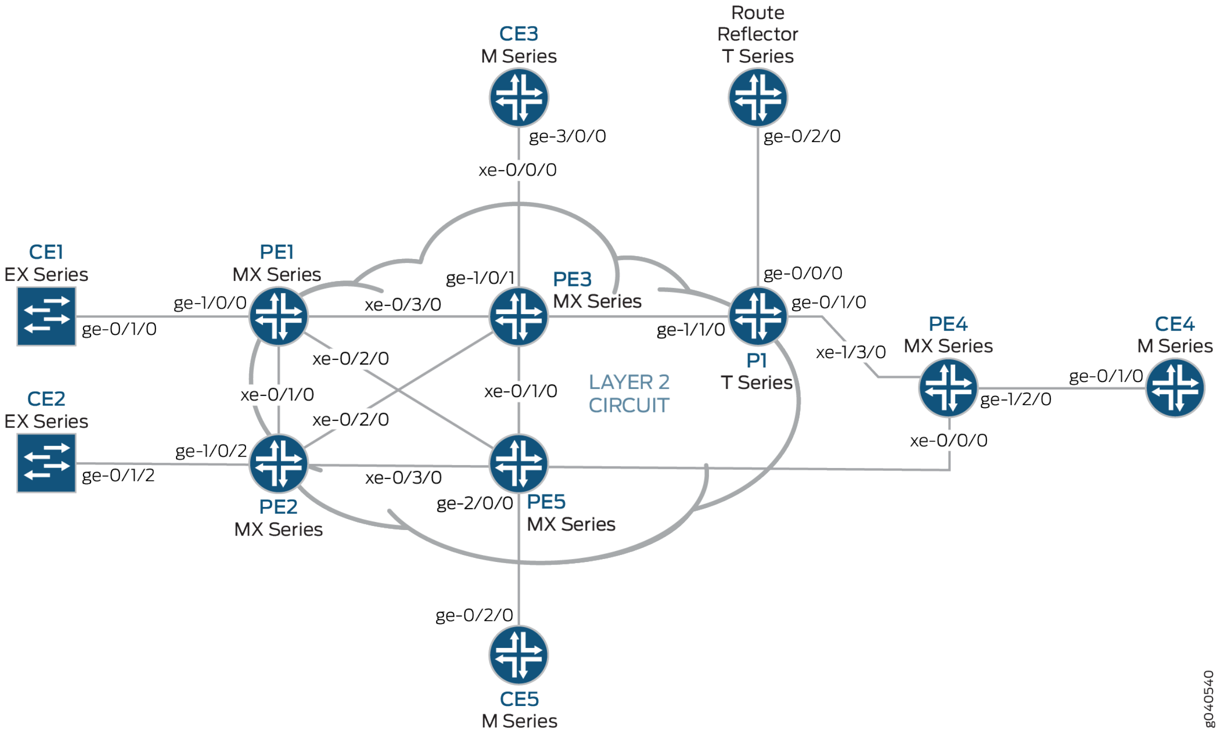
- Junos OS Release 9.3 or later
- 3 MX Series 5G Universal Routing Platforms
- 1 M Series Multiservice Edge Router
- 1 T Series Core Router
- 1 EX Series Ethernet Switch

Overview and Topology

IN THIS SECTION

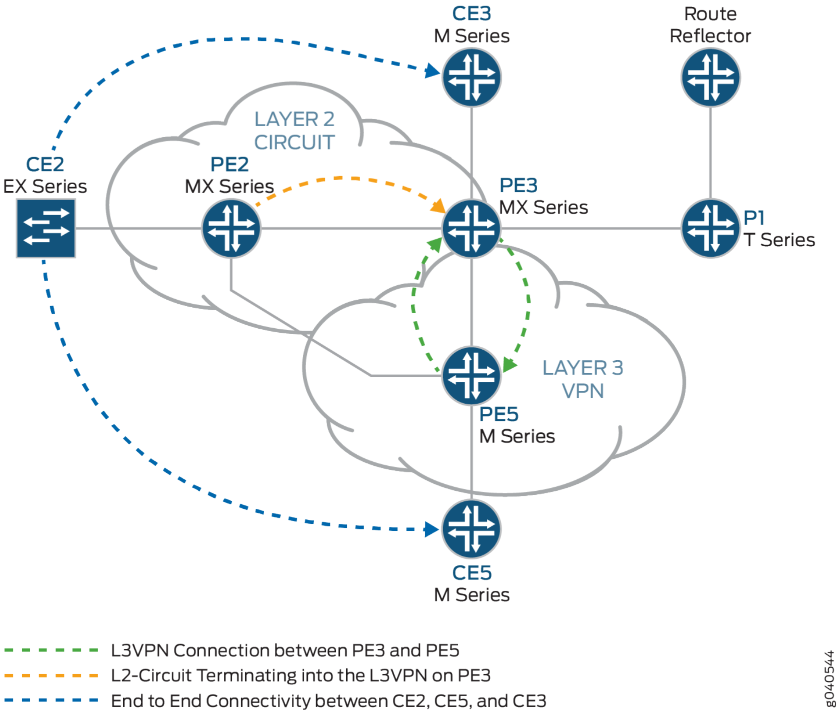
The physical topology of a Layer 2 circuit to Layer 3 VPN interconnection is shown in Figure 42 on page 383.

Figure 42: Physical Topology of a Layer 2 Circuit to Layer 3 VPN Interconnection



The logical topology of a Layer 2 circuit to Layer 3 VPN interconnection is shown in Figure 43 on page 384.

Figure 43: Logical Topology of a Layer 2 Circuit to Layer 3 VPN Interconnection



Topology

Configuration

IN THIS SECTION

NOTE: In any configuration session, it is good practice to verify periodically that the configuration can be committed using the `commit check` command.

In this example, the router being configured is identified using the following command prompts:

- `CE2` identifies the customer edge 2 (CE2) router
- `PE1` identifies the provider edge 1 (PE1) router
- `CE3` identifies the customer edge 3 (CE3) router
- `PE3` identifies the provider edge 3 (PE3) router
- `CE5` identifies the customer edge 5 (CE5) router
- `PE5` identifies the provider edge 5 (PE5) router

This example contains the following procedures:

Configuring PE Router Customer-facing and Loopback Interfaces

Step-by-Step Procedure

To begin building the interconnection, configure the interfaces on the PE routers. If your network contains provider (P) routers, configure the interfaces on the P routers also. This example shows the configuration for Router PE2, Router PE3, and Router PE5.

1. On Router PE2, configure the `ge-1/0/2` interface encapsulation. To configure the interface encapsulation, include the `encapsulation` statement and specify the `ethernet-ccc` option (`vlan-ccc` encapsulation is also supported). Configure the `ge-1/0/2.0` logical interface family for circuit cross-connect functionality. To configure the logical interface family, include the `family` statement and specify the `ccc` option. The encapsulation should be configured the same way for all routers in the Layer 2 circuit domain.

   ```
   [edit interfaces]
   ge-1/0/2 {
       encapsulation ethernet-ccc;
       unit 0 {
           family ccc;
       }
   }
   ```

2. On Router PE2, configure the `lo0.0` interface. Include the `family` statement and specify the `inet` option. Include the `address` statement and specify `192.0.2.2/24` as the loopback IPv4 address.

```
[edit interfaces]
lo0 {
    unit 0 {
        family inet {
            address 192.0.2.2/24;
        }
    }
}
```

3. On Router PE3, configure the `ge-1/0/1` interface. Include the `family` statement and specify the `inet` option. Include the `address` statement and specify `198.51.100.1/24` as the interface address for this device.

```
[edit interfaces]
ge-1/0/1 {
    unit 0 {
        family inet {
            address 198.51.100.1/24;
        }
    }
}
```

4. On Router PE3, configure the `lo0.0` loopback interface. Include the `family` statement and specify the `inet` option. Include the `address` statement and specify `192.0.2.3/24` as the loopback IPv4 address for this router.

```
[edit interfaces]
lo0 {
    unit 0 {
        family inet {
            address 192.0.2.3/24;
        }
    }
}
```

5. On Router PE5, configure the `ge-2/0/0` interface. Include the `family` statement and specify the `inet` option. Include the `address` statement and specify `198.51.100.8/24` as the interface address.

```
[edit interfaces]
ge-2/0/0 {
    unit 0 {
        family inet {
            address 198.51.100.8/24;
        }
    }
}
```

6. On Router PE5, configure the `lo0.0` interface. Include the `family` statement and specify the `inet` option. Include the `address` statement and specify `192.0.2.5/24` as the loopback IPv4 address for this router.

```
[edit interfaces]
lo0 {
    unit 0 {
        family inet {
            address 192.0.2.5/24;
        }
    }
}
```

Configuring Core-facing Interfaces

Step-by-Step Procedure

This procedure describes how to configure the core-facing interfaces on the PE routers. This example does not include all the core-facing interfaces shown in the physical topology illustration. Enable the `mpls` and `inet` address families on the core-facing interfaces.

1. On Router PE2, configure the `xe-0/2/0` interface. Include the `family` statement and specify the `inet` address family. Include the `address` statement and specify `10.10.5.1/30` as the interface address. Include the `family` statement and specify the `mpls` address family.

```
[edit interfaces]
xe-0/2/0 {
    unit 0 {
        family inet {
```

```
            address 10.10.5.1/30;
        }
        family mpls;
    }
}
```

2. On Router PE3, configure the core-facing interfaces. Include the `family` statement and specify the `inet` address family. Include the `address` statement and specify the IPv4 addresses shown in the example as the interface addresses. Include the `family` statement and specify the `mpls` address family. In the example, the `xe-2/1/0` interface is connected to Router PE5, and the `xe-2/2/0` interface is connected to Router PE2.

```
[edit interfaces]
xe-2/0/0 {
    unit 0 {
        family inet {
            address 10.10.20.2/30;
        }
        family mpls;
    }
}
xe-2/1/0 {
    unit 0 {
        family inet {
            address 10.10.6.1/30;
        }
        family mpls;
    }
}
xe-2/2/0 {
    unit 0 {
        family inet {
            address 10.10.5.2/30;
        }
        family mpls;
    }
}
xe-2/3/0 {
    unit 0 {
        family inet {
            address 10.10.1.2/30;
        }
```

```
        family mpls;
    }
}
```

3. On Router PE5, configure the `xe-0/1/0` interface. Include the `family` statement and specify the `inet` address family. Include the `address` statement and specify `10.10.6.2/30` as the interface address. Include the `family` statement and specify the `mpls` address family.

```
[edit interfaces]
xe-0/1/0 {
    unit 0 {
        family inet {
            address 10.10.6.2/30;
        }
        family mpls;
    }
}
```

Configuring Protocols

Step-by-Step Procedure

This procedure describes how to configure the protocols used in this example. If your network contains P routers, configure the interfaces on the P routers also.

1. On Router PE3, enable OSPF as the IGP. Enable the MPLS, LDP, and BGP protocols on all interfaces except `fxp.0`. LDP is used as the signaling protocol for the Layer 2 circuit to Router PE2 . The following configuration snippet shows the protocol configuration for Router PE3:

```
[edit]
protocols {
    rsvp {
        interface all;
        interface fxp0.0 {
            disable;
        }
    }
    mpls {
        label-switched-path to-RR {
            to 192.0.2.7;
```

```
            }
            label-switched-path to-PE2 {
                to 192.0.2.2;
            }
            label-switched-path to-PE5 {
                to 192.0.2.5;
            }
            label-switched-path to-PE4 {
                to 192.0.2.4;
            }
            label-switched-path to-PE1 {
                to 192.0.2.1;
            }
            interface all;
            interface fxp0.0 {
                disable;
            }
        }
        bgp {
            group RR {
                type internal;
                local-address 192.0.2.3;
                family inet-vpn {
                    unicast;
                }
                family l2vpn {
                    signaling;
                }
                neighbor 192.0.2.7;
            }
        }
        ospf {
            traffic-engineering;
            area 0.0.0.0 {
                interface all;
                interface fxp0.0 {
                    disable;
                }
            }
        }
        ldp {
            interface all;
            interface fxp0.0 {
```

```
                disable;
            }
        }
    }
```

2. On Router PE2, configure the MPLS, OSPF, and LDP protocols.

```
[edit ]
protocols {
    mpls {
        interface all;
        interface fxp0.0 {
            disable;
        }
    }
    ospf {
        traffic-engineering;
        area 0.0.0.0 {
            interface all;
            interface fxp0.0 {
                disable;
            }
        }
    }
    ldp {
        interface all;
        interface fxp0.0 {
            disable;
        }
    }
}
```

3. On Router PE5, enable OSPF as the IGP. Enable the MPLS, RSVP, and BGP protocols on all interfaces except `fxp.0`. Enable core-facing interfaces with the `mpls` and `inet` address families.

```
[edit]
protocols {
    rsvp {
        interface all {
            link-protection;
        }
```

```
        interface fxp0.0 {
            disable;
        }
    }
    mpls {
        label-switched-path to-RR {
            to 192.0.2.7;
        }
        label-switched-path to-PE2 {
            to 192.0.2.2;
        }
        label-switched-path to-PE3 {
            to 192.0.2.3;
        }
        label-switched-path to-PE4 {
            to 192.0.2.4;
        }
        label-switched-path to-PE1 {
            to 192.0.2.1;
        }
        interface all;
        interface fxp0.0 {
            disable;
        }
    }
    bgp {
        group to-rr {
            type internal;
            local-address 192.0.2.5;
            family inet-vpn {
                unicast;
            }
            family l2vpn {
                signaling;
            }
            neighbor 192.0.2.7;
        }
    }
    ospf {
        traffic-engineering;
        area 0.0.0.0 {
            interface all;
            interface fxp0.0 {
```

```
                disable;
            }
        }
    }
}
```

Configuring Routing Instances and Layer 2 Circuits

Step-by-Step Procedure

This procedure describes how to configure the Layer 2 circuit and the Layer 3 VPN.

1. On Router PE2, configure the Layer 2 circuit. Include the `l2circuit` statement. Include the `neighbor` statement and specify the loopback IPv4 address of Router PE3 as the neighbor. Include the interface statement and specify `ge-1/0/2.0` as the logical interface that is participating in the Layer 2 circuit. Include the `virtual-circuit-id` statement and specify `100` as the identifier. Include the `no-control-word` statement for equipment that does not support the control word.

```
[edit ]
protocols {
    l2circuit {
        neighbor 192.0.2.3 {
            interface ge-1/0/2.0 {
                virtual-circuit-id 100;
                no-control-word;
            }
        }
    }
}
```

2. On Router PE3, configure the Layer 2 circuit to Router PE2. Include the `l2circuit` statement. Include the `neighbor` statement and specify the loopback IPv4 address of Router PE2 as the neighbor. Include the interface statement and specify `lt-1/1/10.0` as the logical tunnel interface that is participating in the Layer 2 circuit. Include the `virtual-circuit-id` statement and specify `100` as the identifier. Include the `no-control-word` statement.

```
[edit ]
protocols {
    l2circuit {
        neighbor 192.0.2.2 {
```

```
            interface lt-1/1/10.0 {
                virtual-circuit-id 100;
                no-control-word;
            }
        }
    }
}
```

3. On Router PE3, configure the Layer 3 VPN (`L3VPN`) routing instance to Router PE5 at the `[edit routing-instances]` hierarchy level. Also configure the BGP peer group at the `[edit routing-instances L3VPN protocols]` hierarchy level.

```
[edit ]
routing-instances {
    L3VPN {
        instance-type vrf;
        interface ge-1/0/1.0;
        interface lt-1/1/10.1;
        route-distinguisher 65000:33;
        vrf-target target:65000:2;
        vrf-table-label;
        protocols {
            bgp {
                export direct;
                group ce3 {
                    neighbor 198.51.100.6{
                        peer-as 100;
                    }
                }
            }
        }
    }
}
```

4. On Router PE5, configure the Layer 3 VPN routing instance (`L3VPN`) at the `[edit routing-instances]` hierarchy level. Also configure the BGP peer group at the `[edit routing-instances L3VPN protocols]` hierarchy level.

```
[edit ]
routing-instances {
    L3VPN {
```

```
        instance-type vrf;
        interface ge-2/0/0.0;
        route-distinguisher 65000:5;
        vrf-target target:65000:2;
        vrf-table-label;
        protocols {
            bgp {
                group ce5 {
                    neighbor 198.51.100.10 {
                        peer-as 200;
                    }
                }
            }
        }
    }
}
```

Configuring the Route Reflector

Step-by-Step Procedure

Although a route reflector is not required to interconnect a Layer 2 circuit with a Layer 3 VPN, this examples uses a route reflector. This procedure shows the relevant portion of the route reflector configuration.

1. Configure the route reflector with RSVP, MPLS, BGP and OSPF. The route reflector is a BGP peer with the PE routers. Notice that the BGP peer group configuration includes the `family` statement and specifies the `inet-vpn` option The `inet-vpn` option enables BGP to advertise network layer reachability information (NLRI) for the Layer 3 VPN routes. The configuration also includes the `family` statement and specifies the `l2vpn` option. The `l2vpn` option enables BGP to advertise NLRI for the Layer 2 circuit. Layer 2 circuits use the same internal BGP infrastructure as Layer 2 VPNs.

```
[edit ]
protocols {
    rsvp {
        interface all;
        interface fxp0.0 {
            disable;
        }
    }
    mpls {
```

```
        label-switched-path to-pe3 {
            to 192.0.2.3;
        }
        label-switched-path to-pe5 {
            to 192.0.2.5;
        }
        interface all;
        interface fxp0.0 {
            disable;
        }
    }
    bgp {
        group RR {
            type internal;
            local-address 192.0.2.7;
            family inet {
                unicast;
            }
            family inet-vpn {
                unicast;
            }
            family l2vpn {
                signaling;
            }
            cluster 192.0.2.7;
            neighbor 192.0.2.1;
            neighbor 192.0.2.2;
            neighbor 192.0.2.4;
            neighbor 192.0.2.5;
            neighbor 192.0.2.3;
        }
    }
    ospf {
        traffic-engineering;
        area 0.0.0.0 {
            interface all;
            interface fxp0.0 {
                disable;
            }
        }
    }
}
```

Interconnecting the Layer 2 Circuit with the Layer 3 VPN

Step-by-Step Procedure

Before you can configure the logical tunnel interface in an MX Series router, you must create the tunnel services interface to be used for tunnel services.

1. Create the tunnel service interface on Router PE3. Include the `bandwidth` statement at the `[edit chassis fpc` *`slot-number`* `pic` *`slot-number`* `tunnel-services]` hierarchy level and specify the amount of bandwidth to reserve for tunnel services in gigabits per second.

   ```
   [edit chassis]
   fpc 1 {
       pic 1 {
           tunnel-services {
               bandwidth 1g;
           }
       }
   }
   ```

2. On Router PE3, configure the `lt-1/1/10` logical tunnel interface unit 0.

   Router PE3 is the router that is *stitching* the Layer 2 circuit to the Layer 3 VPN using the logical tunnel interface. The configuration of the peer unit interfaces is what makes the interconnection.

   Include the `encapsulation` statement and specify the `ethernet-ccc` option. Include the `peer-unit` statement and specify the logical interface unit `1` as the peer tunnel interface. Include the `family` statement and specify the `ccc` option.

   Configure the `lt-1/1/10` logical interface unit `1` with `ethernet` encapsulation. Include the `peer-unit` statement and specify the logical interface unit `0` as the peer tunnel interface. Include the `family` statement and specify the `inet` option. Also include the `address` statement and specify `198.51.100.11/24` as the IPv4 address of the interface.

   NOTE: The peering logical interfaces must belong to the same logical tunnel interface derived from the Tunnel Services PIC.

   ```
   [edit interfaces]
   lt-1/1/10 {
       unit 0 {
           encapsulation ethernet-ccc;
   ```

```
        peer-unit 1;
        family ccc;
    }
    unit 1 {
        encapsulation ethernet;
        peer-unit 0;
        family inet {
            address 198.51.100.11/24;
        }
    }
}
```

3. On each router, commit the configuration.

```
user@host> commit check
configuration check succeeds
user@host> commit
```

Verifying the Layer 2 Circuit to Layer 3 VPN Interconnection

IN THIS SECTION

To verify that the interconnection is working properly, perform these tasks:

Verifying That the Layer 2 Circuit Connection to Router PE3 is Up

Purpose

To verify that the Layer 2 circuit connection from Router PE2 to Router PE3 is `Up`. To also document the incoming and outgoing LDP labels and the circuit ID used by this Layer 2 circuit connection.

Action

Verify that the Layer 2 circuit connection is up, using the `show l2circuit connections` command.

```
user@PE2> show l2circuit connections
Legend for connection status (St)
EI -- encapsulation invalid      NP -- interface h/w not present
MM -- mtu mismatch               Dn -- down
EM -- encapsulation mismatch     VC-Dn -- Virtual circuit Down
CM -- control-word mismatch      Up -- operational
VM -- vlan id mismatch           CF -- Call admission control failure
OL -- no outgoing label          IB -- TDM incompatible bitrate
NC -- intf encaps not CCC/TCC    TM -- TDM misconfiguration
BK -- Backup Connection          ST -- Standby Connection
CB -- rcvd cell-bundle size bad  SP -- Static Pseudowire
LD -- local site signaled down   RS -- remote site standby
RD -- remote site signaled down  XX -- unknown

Legend for interface status
Up -- operational
Dn -- down
Neighbor: 192.0.2.3
    Interface                 Type  St     Time last up          # Up trans
    ge-1/0/2.0(vc 100)        rmt   Up     Jan  7 02:14:13 2010           1
      Remote PE: 192.0.2.3, Negotiated control-word: No
      Incoming label: 301488, Outgoing label: 315264
      Negotiated PW status TLV: No
      Local interface: ge-1/0/2.0, Status: Up, Encapsulation: ETHERNET
```

Meaning

The output shows that the Layer 2 circuit connection from Router PE2 to Router PE3 is `Up` and the connection is using the `ge-1/0/2.0` interface. Note that the outgoing label is `315264` and the incoming label is `301488`, the virtual circuit (VC) identifier is `100` and the encapsulation is `ETHERNET`.

Verifying LDP Neighbors and Targeted LDP LSPs on Router PE2

Purpose

To verify that Router PE2 has a targeted LDP LSP to Router PE3 and that Router PE2 and Router PE3 are LDP neighbors.

Action

Verify that Router PE2 has a targeted LDP LSP to Router PE3 and that Router PE2 and Router PE3 are LDP neighbors, using the `show ldp neighbor` command.

```
user@PE2> show ldp neighbor
Address            Interface          Label space ID         Hold time
192.0.2.3            lo0.0              192.0.2.3:0             38
```

Meaning

The output shows that Router PE2 has an LDP neighbor with the IPv4 address of `192.0.2.3`. Address 192.0.2.3 is the lo0.0 interface address of Router PE3. Notice that Router PE2 uses the local `lo0.0` interface for the LSP.

Verifying that the routers are LDP neighbors also verifies that the targeted LSP is established.

Verifying the Layer 2 Circuit Routes on Router PE2

Purpose

To verify that Router PE2 has a route for the Layer 2 circuit and that the route uses the LDP MPLS label to Router PE3.

Action

Verify that Router PE2 has a route for the Layer 2 circuit and that the route uses the LDP MPLS label to Router PE3, using the `show route table mpls.0` command.

```
user@PE2> show route table mpls.0
mpls.0: 13 destinations, 13 routes (13 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both
```

```
0                  *[MPLS/0] 1w3d 05:24:11, metric 1
                      Receive
1                  *[MPLS/0] 1w3d 05:24:11, metric 1
                      Receive
2                  *[MPLS/0] 1w3d 05:24:11, metric 1
                      Receive
300560             *[LDP/9] 16:12:23, metric 1
                    > to 10.10.2.1 via xe-0/1/0.0, Pop
300560(S=0)        *[LDP/9] 16:12:23, metric 1
                    > to 10.10.2.1 via xe-0/1/0.0, Pop
301008             *[LDP/9] 16:12:23, metric 1
                    > to 10.10.4.2 via xe-0/3/0.0, Swap 299856
301488             *[L2CKT/7] 11:07:28
                    > via ge-1/0/2.0, Pop
301536             *[LDP/9] 16:12:23, metric 1
                    > to 10.10.4.2 via xe-0/3/0.0, Pop
301536(S=0)        *[LDP/9] 16:12:23, metric 1
                    > to 10.10.4.2 via xe-0/3/0.0, Pop
301712             *[LDP/9] 12:41:22, metric 1
                    > to 10.10.5.2 via xe-0/2/0.0, Swap 315184
301728             *[LDP/9] 12:41:22, metric 1
                    > to 10.10.5.2 via xe-0/2/0.0, Pop
301728(S=0)        *[LDP/9] 12:41:22, metric 1
                    > to 10.10.5.2 via xe-0/2/0.0, Pop
ge-1/0/2.0         *[L2CKT/7] 11:07:28, metric2 1
                    > to 10.10.5.2 via xe-0/2/0.0, Push 315264
```

Meaning

The output shows that Router PE2 pushes the `315264` outgoing label on the `L2CKT` route going out interface `ge-1/0/2.0`. The output also shows that Router PE2 pops the `301488` incoming label on the `L2CKT` coming from interface `ge-1/0/2.0`

Verifying That the Layer 2 Circuit Connection to Router PE2 is Up

Purpose

To verify that the Layer 2 circuit connection from Router PE3 to Router PE2 is `Up`, To also document the incoming and outgoing LDP labels and the circuit ID used by this Layer 2 circuit connection.

Action

Verify that the Layer 2 circuit connection is up, using the `show l2circuit connections` command.

```
user@PE3> show l2circuit connections
Layer-2 Circuit Connections:
Legend for connection status (St)
EI -- encapsulation invalid      NP -- interface h/w not present
MM -- mtu mismatch               Dn -- down
EM -- encapsulation mismatch     VC-Dn -- Virtual circuit Down
CM -- control-word mismatch      Up -- operational
VM -- vlan id mismatch           CF -- Call admission control failure
OL -- no outgoing label          IB -- TDM incompatible bitrate
NC -- intf encaps not CCC/TCC    TM -- TDM misconfiguration
BK -- Backup Connection          ST -- Standby Connection
CB -- rcvd cell-bundle size bad  XX -- unknown

Legend for interface status
Up -- operational
Dn -- down
Neighbor: 192.0.2.2
    Interface                 Type  St     Time last up          # Up trans
    lt-1/1/10.0(vc 100)       rmt   Up     Jan  7 02:15:03 2010           1
      Remote PE: 192.0.2.2, Negotiated control-word: No
      Incoming label: 315264, Outgoing label: 301488
      Local interface: lt-1/1/10.0, Status: Up, Encapsulation: ETHERNET
```

Meaning

The output shows that the Layer 2 circuit connection from Router PE3 to Router PE2 is `Up` and the connection is using the logical tunnel (`lt`) interface. Note that the incoming label is `315264` and the outgoing label is `301488`, the virtual circuit (VC) identifier is `100`, and that the encapsulation is `ETHERNET`.

Verifying LDP Neighbors and Targeted LDP LSPs on Router PE3

Purpose

To verify that Router PE3 has a targeted LDP LSP to Router PE2 and that Router PE3 and Router PE2 are LDP neighbors.

Action

Verify that Router PE2 has a targeted LDP LSP to Router PE3 and that Router PE2 and Router PE3 are LDP neighbors, using the `show ldp neighbor` command.

```
user@PE2> show ldp neighbor
Address            Interface          Label space ID         Hold time
192.0.2.2            lo0.0              192.0.2.2:0               43
192.0.2.4            lo0.0              192.0.2.4:0               33
```

Meaning

The output shows that Router PE3 has an LDP neighbor with the IPv4 address of `192.0.2.2`. Address 192.0.2.2 is the lo0.0 interface address of Router PE2. The output also shows that the interface used on Router PE3 for the LSP is `lo0.0`. Verifying that the routers are LDP neighbors also verifies that the targeted LSP is established.

Verifying a BGP Peer Session with the Route Reflector on Router PE3

Purpose

To verify that Router PE3 has a peer session established with the route reflector.

Action

Verify that Router PE3 has a peer session established with the route reflector, using the `show bgp summary` command.

```
user@PE2> show bgp summary
Groups: 2 Peers: 2 Down peers: 0
Table          Tot Paths  Act Paths Suppressed    History Damp State    Pending
bgp.l3vpn.0            1          1          0          0          0          0
Peer                     AS      InPkt     OutPkt    OutQ   Flaps Last Up/Dwn State|#Active/
Received/Accepted/Damped...
192.0.2.7              65000       1597       1612       0       1    12:03:21 Establ
  bgp.l2vpn.0: 0/0/0/0
  bgp.l3vpn.0: 1/1/1/0
  L3VPN.inet.0: 1/1/1/0
```

Meaning

The output shows that Router PE3 has a peer session with the router with the IPv4 address of `192.0.2.7`. Address 192.0.2.7 is the lo0.0 interface address of the route reflector. The output also shows that the peer session state is `Establ`, meaning that the session is established.

Verifying the Layer 3 VPN Routes on Router PE3

Purpose

To verify that Router PE3 has Layer 3 VPN routes to Router CE2, Router CE3, and Router CE5.

Action

Verify that Router PE3 has routes to Router CE2, Router CE3, and Router CE5 in the Layer 3 VPN route table, using the `show route table L3VPN.inet.0` command. In this example, `L3VPN` is the name configured for the routing instance.

```
user@PE3> show route table L3VPN.inet.0
L3VPN.inet.0: 5 destinations, 5 routes (5 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

198.51.100.10/24      *[Direct/0] 11:13:59
                    > via lt-1/1/10.1
198.51.100.11/24      *[Local/0] 11:13:59
                      Local via lt-1/1/10.1
198.51.100.12/24      *[BGP/170] 11:00:41, localpref 100, from 192.0.2.7
                      AS path: I
                    > to 10.10.6.2 via xe-2/1/0.0, Push 16
198.51.100.13/24      *[Direct/0] 11:54:41
                    > via ge-1/0/1.0
198.51.100.1/24      *[Local/0] 11:54:41
                      Local via ge-1/0/1.0
```

Meaning

The output shows that Router PE3 has a route to the IPv4 subnetwork address of `198.51.100.10`. Address 198.51.100.15 is the interface address of Router CE2. The output shows that Router PE3 has a route to the IPv4 subnetwork address of `198.51.100.12`. Address 198.51.100.10 is the interface address of Router CE5. The output shows that Router PE3 has a route to the IPv4 subnetwork address of `198.51.100.13`. Address 198.51.100.6 is the interface address of Router CE3.

Verifying the Layer 2 Circuit Routes on Router PE3

Purpose

To verify that Router PE3 has a route to Router PE2 in the Layer 2 circuit route table.

Action

Verify that Router PE3 has a route to Router PE2 in the Layer 2 circuit route table, using the `show route table l2circuit.0` command.

```
user@PE3> show route table l2circuit.0
192.0.2.2:NoCtrlWord:5:100:Local/96 (1 entry, 1 announced)
        *L2CKT  Preference: 7
                Next hop type: Indirect
                Next-hop reference count: 1
                Next hop type: Router
                Next hop: 10.10.5.1 via xe-2/2/0.0, selected
                Protocol next hop: 192.0.2.2
                Indirect next hop: 8cae0a0 -
                State: <Active Int>
                Local AS: 65000
                Age: 11:16:50   Metric2: 1
                Task: l2 circuit
                Announcement bits (1): 0-LDP
                AS path: I
                VC Label 315264, MTU 1500
```

Meaning

The output shows that Router PE3 has a route to the IPv4 address of `192.0.2.2`. Address 192.0.2.2 is the lo0.0 interface address of Router PE2. Note that the VC label is `315264`. This label is the same as the incoming MPLS label displayed using the `show l2circuit connections` command.

Verifying the MPLS Routes on Router PE3

Purpose

To verify that Router PE3 has a route to Router PE2 in the MPLS route table.

Action

Verify Router PE3 has a route to Router PE2 in the MPLS route table, using the `show route table mpls.0` command.

```
user@PE3> show route table mpls.0
mpls.0: 21 destinations, 21 routes (21 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

0                  *[MPLS/0] 1w3d 05:29:02, metric 1
                      Receive
1                  *[MPLS/0] 1w3d 05:29:02, metric 1
                      Receive
2                  *[MPLS/0] 1w3d 05:29:02, metric 1
                      Receive
16                 *[VPN/0] 12:22:45
                      to table L3VPN.inet.0, Pop
315184             *[LDP/9] 12:45:14, metric 1
                    > to 10.10.20.1 via xe-2/0/0.0, Pop
315184(S=0)        *[LDP/9] 12:45:14, metric 1
                    > to 10.10.20.1 via xe-2/0/0.0, Pop
315200             *[LDP/9] 00:03:53, metric 1
                    > to 10.10.20.1 via xe-2/0/0.0, Swap 625297
                      to 10.10.6.2 via xe-2/1/0.0, Swap 299856
315216             *[LDP/9] 12:45:14, metric 1
                    > to 10.10.6.2 via xe-2/1/0.0, Pop
315216(S=0)        *[LDP/9] 12:45:14, metric 1
                    > to 10.10.6.2 via xe-2/1/0.0, Pop
315232             *[LDP/9] 12:45:06, metric 1
                    > to 10.10.1.1 via xe-2/3/0.0, Pop
315232(S=0)        *[LDP/9] 12:45:06, metric 1
                    > to 10.10.1.1 via xe-2/3/0.0, Pop
315248             *[LDP/9] 12:45:14, metric 1
                    > to 10.10.5.1 via xe-2/2/0.0, Pop
315248(S=0)        *[LDP/9] 12:45:14, metric 1
                    > to 10.10.5.1 via xe-2/2/0.0, Pop
315264             *[L2CKT/7] 11:11:20
                    > via lt-1/1/10.0, Pop
315312             *[RSVP/7] 11:26:01, metric 1
                    > to 10.10.6.2 via xe-2/1/0.0, label-switched-path to-pe5
315312(S=0)        *[RSVP/7] 11:26:01, metric 1
                    > to 10.10.6.2 via xe-2/1/0.0, label-switched-path to-pe5
```

```
315328             *[RSVP/7] 11:26:01, metric 1
                    > to 10.10.20.1 via xe-2/0/0.0, label-switched-path to-RR
315360             *[RSVP/7] 11:26:01, metric 1
                    > to 10.10.20.1 via xe-2/0/0.0, label-switched-path to-RR
316208             *[RSVP/7] 00:03:32, metric 1
                    > to 10.10.6.2 via xe-2/1/0.0, label-switched-path Bypass->10.10.9.1
316208(S=0)        *[RSVP/7] 00:03:32, metric 1
                    > to 10.10.6.2 via xe-2/1/0.0, label-switched-path Bypass->10.10.9.1
lt-1/1/10.0        *[L2CKT/7] 11:11:20, metric2 1
                    > to 10.10.5.1 via xe-2/2/0.0, Push 301488
```

Meaning

The output shows that Router PE3 has a route for the Layer 2 circuit and that the route uses the LDP MPLS label to Router PE2. Notice that the `301488` label is the same as the outgoing label displayed on Router PE2 using the `show l2circuit connections` command.

Verifying Traffic Flow Between Router CE2 and Router CE3

Purpose

To verify that the CE routers can send and receive traffic across the interconnection.

Action

Verify that Router CE2 can send traffic to and receive traffic from Router CE3 across the interconnection, using the `ping` command.

```
user@CE2>ping 198.51.100.6
PING 198.51.100.6 (198.51.100.6): 56 data bytes
64 bytes from 198.51.100.6: icmp_seq=0 ttl=63 time=0.708 ms
64 bytes from 198.51.100.6: icmp_seq=1 ttl=63 time=0.610 ms
```

Meaning

The output shows that Router CE2 can send an ICMP request to and receive a response from Router CE3 across the interconnection.

Verifying Traffic Flow Between Router CE2 and Router CE5

Purpose

To verify that the CE routers can send and receive traffic across the interconnection.

Action

Verify that Router CE2 can send traffic to and receive traffic from Router CE5 across the interconnection, using the `ping` command.

```
user@CE2>ping 198.51.100.10
PING 198.51.100.10 (198.51.100.10): 56 data bytes
64 bytes from 198.51.100.10: icmp_seq=0 ttl=62 time=0.995 ms
64 bytes from 198.51.100.10: icmp_seq=1 ttl=62 time=1.005 ms
```

Meaning

The output shows that Router CE2 can send an ICMP request to and receive a response from Router CE5 across the interconnection.

RELATED DOCUMENTATION

- Understanding Layer 3 VPNs | 6
- Pseudowire Subscriber Logical Interfaces Overview
- Configuring a Pseudowire Subscriber Logical Interface

Connecting Layer 3 VPNs to Layer 2 VPNs

IN THIS SECTION

Interconnecting Layer 2 VPNs with Layer 3 VPNs Overview

IN THIS SECTION

As MPLS-based Layer 2 services grow in demand, new challenges arise for service providers to be able to interoperate with Layer 2 and Layer 3 services and give their customers value-added services. Junos OS has various features to address the needs of service providers. One of these features is the use of a logical tunnel interface. This Junos OS functionality makes use of a tunnel PIC to loop packets out and back from the Packet Forwarding Engine to link the Layer 2 network with the Layer 3 network. The solution is limited by the logical tunnel bandwidth constraints imposed by the tunnel PIC.

Interconnecting Layer 2 VPNs with Layer 3 VPNs Applications

Interconnecting a Layer 2 VPN with a Layer 3 VPN provides the following benefits:

- A single access line to provide multiple services—Traditional VPNs over Layer 2 circuits require the provisioning and maintenance of separate networks for IP and for VPN services. In contrast, Layer 2 VPNs enable the sharing of a provider's core network infrastructure between IP and Layer 2 VPN services, thereby reducing the cost of providing those services.

- Flexibility—Many different types of networks can be accommodated by the service provider. If all sites in a VPN are owned by the same enterprise, this is an intranet. If various sites are owned by different enterprises, the VPN is an extranet. A site can be located in more than one VPN.

- Wide range of possible policies—You can give every site in a VPN a different route to every other site, or you can force traffic between certain pairs of sites routed via a third site and so pass certain traffic through a firewall.

- Scalable network—This design enhances the scalability because it eliminates the need for provider edge (PE) routers to maintain all of the service provider's VPN routes. Each PE router maintains a VRF table for each of its directly connected sites. Each customer connection (such as a Frame Relay PVC, an ATM PVC, or a VLAN) is mapped to a specific VRF table. Thus, it is a port on the PE router and not a site that is associated with a VRF table. Multiple ports on a PE router can be associated with a single VRF table. It is the ability of PE routers to maintain multiple forwarding tables that supports the per-VPN segregation of routing information.

- Use of route reflectors—Provider edge routers can maintain IBGP sessions to route reflectors as an alternative to a full mesh of IBGP sessions. Deploying multiple route reflectors enhances the scalability of the RFC 2547bis model because it eliminates the need for any single network component to maintain all VPN routes.

- Multiple VPNs are kept separate and distinct from each other—The customer edge routers do not peer with each other. Two sites have IP connectivity over the common backbone only, and only if there is a VPN which contains both sites. This feature keeps the VPNs separate and distinct from each other, even if two VPNs have an overlapping address space.
- Simple for customers to use—Customers can obtain IP backbone services from a service provider, and they do not need to maintain their own backbones.

Example: Interconnecting a Layer 2 VPN with a Layer 3 VPN

IN THIS SECTION

This example provides a step-by-step procedure and commands for interconnecting and verifying a Layer 2 VPN with a Layer 3 VPN. It contains the following sections:

Requirements

This example uses the following hardware and software components:

- Junos OS Release 9.3 or later
- Five MX Series routers
- Three M Series routers
- Two T Series routers

Overview and Topology

IN THIS SECTION

A Layer 2 VPN is a type of virtual private network (VPN) that uses MPLS labels to transport data. The communication occurs between the provider edge (PE) routers.

Layer 2 VPNs use BGP as the signaling protocol and, consequently, have a simpler design and require less provisioning overhead than traditional VPNs over Layer 2 circuits. BGP signaling also enables autodiscovery of Layer 2 VPN peers. Layer 2 VPNs can have either a full-mesh or a hub-and-spoke topology. The tunneling mechanism in the core network is, typically, MPLS. However, Layer 2 VPNs can also use other tunneling protocols, such as GRE.

Layer 3 VPNs are based on RFC 2547bis, *BGP/MPLS IP VPNs*. RFC 2547bis defines a mechanism by which service providers can use their IP backbones to provide VPN services to their customers. A Layer 3 VPN is a set of sites that share common routing information and whose connectivity is controlled by a collection of policies. The sites that make up a Layer 3 VPN are connected over a provider's existing public Internet backbone. RFC 2547bis VPNs are also known as BGP/MPLS VPNs because BGP is used to distribute VPN routing information across the provider's backbone, and MPLS is used to forward VPN traffic across the backbone to remote VPN sites.

Customer networks, because they are private, can use either public addresses or private addresses, as defined in RFC 1918, *Address Allocation for Private Internets*. When customer networks that use private addresses connect to the public Internet infrastructure, the private addresses might overlap with the same private addresses used by other network users. MPLS/BGP VPNs solve this problem by adding a *route distinguisher*. A route distinguisher is a VPN identifier prefix that is added to each address from a particular VPN site, thereby creating an address that is unique both within the VPN and within the Internet.

In addition, each VPN has its own VPN-specific routing table that contains the routing information for that VPN only. To separate a VPN's routes from routes in the public Internet or those in other VPNs, the PE router creates a separate routing table for each VPN called a VPN routing and forwarding (VRF) table. The PE router creates one VRF table for each VPN that has a connection to a customer edge (CE) router. Any customer or site that belongs to the VPN can access only the routes in the VRF tables for that VPN. Every VRF table has one or more extended community attributes associated with it that identify the route as belonging to a specific collection of routers. One of these, the *route target* attribute, identifies a collection of sites (VRF tables) to which a PE router distributes routes. The PE router uses the route target to constrain the import of remote routes into its VRF tables.

When an ingress PE router receives routes advertised from a directly connected CE router, it checks the received route against the VRF export policy for that VPN.

- If it matches, the route is converted to VPN-IPv4 format—that is, the route distinguisher is added to the route. The PE router then announces the route in VPN-IPv4 format to the remote PE routers. It also attaches a route target to each route learned from the directly connected sites. The route target attached to the route is based on the value of the VRF table's configured export target policy. The routes are then distributed using IBGP sessions, which are configured in the provider's core network.

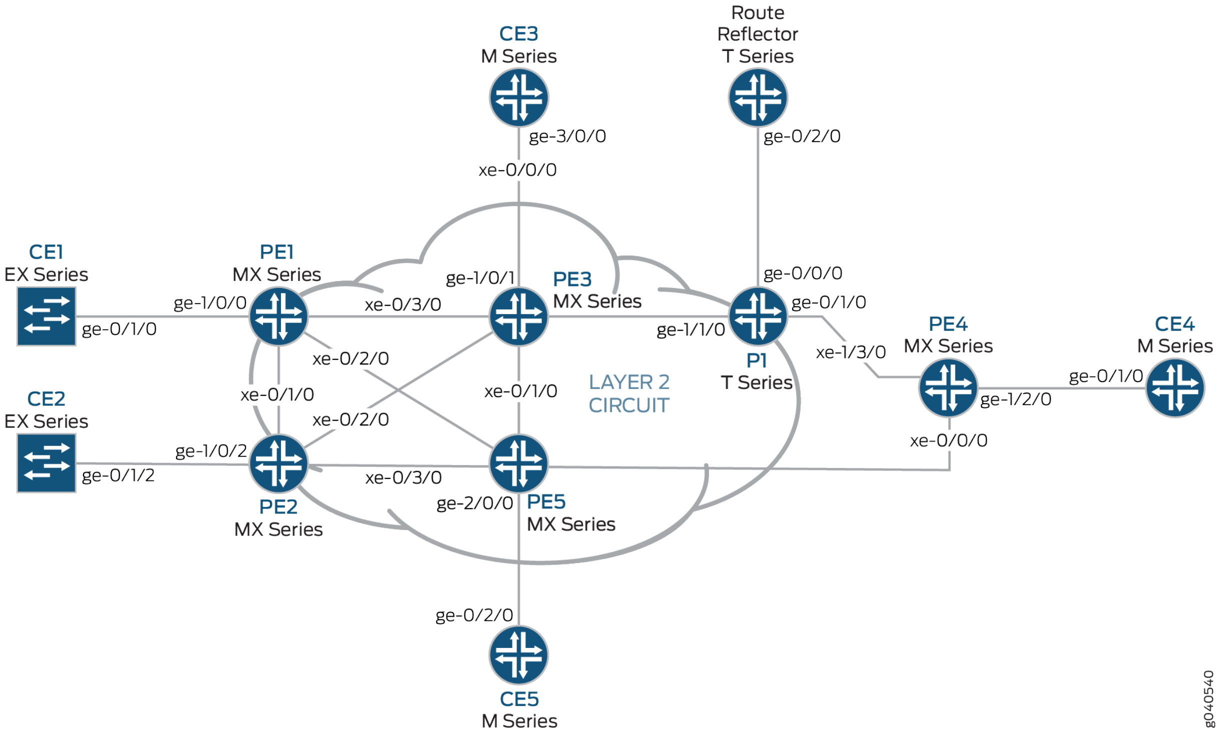
- If the route from the CE router does not match, it is not exported to other PE routers, but it can still be used locally for routing, for example, if two CE routers in the same VPN are directly connected to the same PE router.

When an egress PE router receives a route, it checks it against the import policy on the IBGP session between the PE routers. If it is accepted, the router places the route into its bgp.l3vpn.0 table. At the same time, the router checks the route against the VRF import policy for the VPN. If it matches, the route distinguisher is removed from the route and the route is placed into the VRF table (the *routing-instance-name*.inet.0 table) in IPv4 format.

Topology

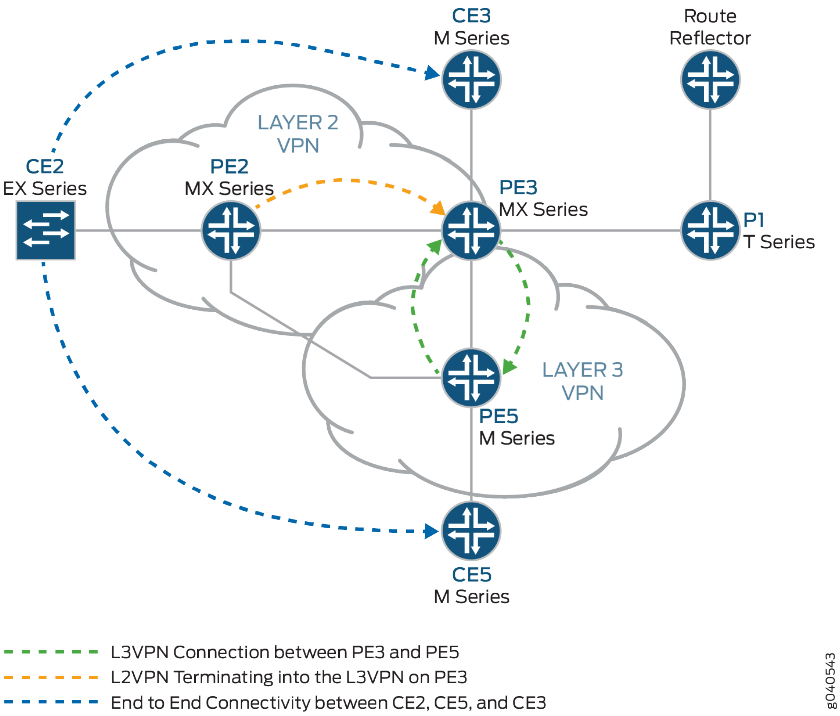
Figure 44 on page 413 shows the physical topology of a Layer 2 VPN-to-Layer 3 VPN interconnection.

Figure 44: Physical Topology of a Layer 2 VPN Terminating into a Layer 3 VPN



The logical topology of a Layer 2 VPN-to-Layer 3 VPN interconnection is shown in Figure 45 on page 414.

Figure 45: Logical Topology of a Layer 2 VPN Terminating into a Layer 3 VPN



The following definitions describe the meaning of the device abbreviations used in Figure 44 on page 413 and Figure 45 on page 414.

- Customer edge (CE) device—A device at the customer premises that provides access to the service provider’s VPN over a data link to one or more provider edge (PE) routers.

  Typically the CE device is an IP router that establishes an adjacency with its directly connected PE routers. After the adjacency is established, the CE router advertises the site’s local VPN routes to the PE router and learns remote VPN routes from the PE router.

- Provider edge (PE) device—A device, or set of devices, at the edge of the provider network that presents the provider's view of the customer site.

  PE routers exchange routing information with CE routers. PE routers are aware of the VPNs that connect through them, and PE routers maintain VPN state. A PE router is only required to maintain VPN routes for those VPNs to which it is directly attached. After learning local VPN routes from CE routers, a PE router exchanges VPN routing information with other PE routers using IBGP. Finally, when using MPLS to forward VPN data traffic across the provider’s backbone, the ingress PE router functions as the ingress label-switching router (LSR) and the egress PE router functions as the egress LSR.

- Provider (P) device—A device that operates inside the provider's core network and does not directly interface to any CE.

Although the P device is a key part of implementing VPNs for the service provider’s customers and may provide routing for many provider-operated tunnels that belong to different VPNs, it is not itself VPN-aware and does not maintain VPN state. Its principal role is allowing the service provider to scale its VPN offerings, for example, by acting as an aggregation point for multiple PE routers.

P routers function as MPLS transit LSRs when forwarding VPN data traffic between PE routers. P routers are required only to maintain routes to the provider’s PE routers; they are not required to maintain specific VPN routing information for each customer site.

Configuration

IN THIS SECTION

To interconnect a Layer 2 VPN with a Layer 3 VPN, perform these tasks:

Configuring the Base Protocols and Interfaces

Step-by-Step Procedure

1. On each PE and P router, configure OSPF with traffic engineering extensions on all interfaces. Disable OSPF on the fxp0.0 interface.

```
[edit protocols]
ospf {
    traffic-engineering;
    area 0.0.0.0 {
        interface all;
        interface fxp0.0 {
            disable;
        }
    }
}
```

2. On all the core routers, enable MPLS on all interfaces. Disable MPLS on the fxp0.0 interface.

```
[edit protocols]
mpls {
    interface all;
    interface fxp0.0 {
        disable;
    }
}
```

3. On all the core routers, create an internal BGP peer group and specify the route reflector address (192.0.2.7) as the neighbor. Also enable BGP to carry Layer 2 VPLS network layer reachability information (NLRI) messages for this peer group by including the `signaling` statement at the `[edit protocols bgp group` *`group-name`* `family l2vpn]` hierarchy level.

```
[edit protocols]
bgp {
    group RR {
        type internal;
        local-address 192.0.2.2;
        family l2vpn {
            signaling;
        }
        neighbor 192.0.2.7;
    }
}
```

4. On Router PE3, create an internal BGP peer group and specify the route reflector IP address (192.0.2.7) as the neighbor. Enable BGP to carry Layer 2 VPLS NLRI messages for this peer group and enable the processing of VPN-IPv4 addresses by including the `unicast` statement at the `[edit protocols bgp group` *`group-name`* `family inet-vpn]` hierarchy level.

```
[edit protocols]
bgp {
    group RR {
        type internal;
        local-address 192.0.2.3;
        family inet-vpn {
            unicast;
        }
```

```
        family l2vpn {
            signaling;
        }
        neighbor 192.0.2.7;
    }
}
```

5. For the Layer 3 VPN domain on Router PE3 and Router PE5, enable RSVP on all interfaces. Disable RSVP on the fxp0.0 interface.

```
[edit protocols]
rsvp {
    interface all;
    interface fxp0.0 {
        disable;
    }
}
```

6. On Router PE3 and Router PE5, create label-switched paths (LSPs) to the route reflector and the other PE routers. The following example shows the configuration on Router PE5.

```
[edit protocols]
mpls {
    label-switched-path to-RR {
        to 192.0.2.7;
    }
    label-switched-path to-PE2 {
        to 192.0.2.2;
    }
    label-switched-path to-PE3 {
        to 192.0.2.3;
    }
    label-switched-path to-PE4 {
        to 192.0.2.4;
    }
    label-switched-path to-PE1 {
        to 192.0.2.1;
    }
}
```

7. On Routers PE1, PE2, PE3, and PE5, configure the core interfaces with an IPv4 address and enable the MPLS address family. The following example shows the configuration of the xe-0/1/0 interface on Router PE2.

```
[edit]
interfaces {
    xe-0/1/0 {
        unit 0 {
            family inet {
                address 10.10.2.2/30;
            }
            family mpls;
        }
    }
}
```

8. On Router PE2 and Router PE3, configure LDP for the Layer 2 VPN MPLS signaling protocol for all interfaces. Disable LDP on the fxp0.0 interface. (RSVP can also be used.)

```
[edit protocols]
ldp {
    interface all;
    interface fxp0.0 {
        disable;
    }
}
```

9. On the route reflector, create an internal BGP peer group and specify the PE routers IP addresses as the neighbors.

```
[edit]
protocols {
    bgp {
        group RR {
            type internal;
            local-address 192.0.2.7;
            family inet {
                unicast;
            }
            family inet-vpn {
```

```
                unicast;
            }
            family l2vpn {
                signaling;
            }
            cluster 192.0.2.7;
            neighbor 192.0.2.1;
            neighbor 192.0.2.2;
            neighbor 192.0.2.4;
            neighbor 192.0.2.5;
            neighbor 192.0.2.3;
        }
    }
}
```

10. On the route reflector, configure MPLS LSPs towards Routers PE3 and PE5 to resolve the BGP next hops from inet.3 routing table.

```
[edit]
protocols {
    mpls {
        label-switched-path to-pe3 {
            to 192.0.2.3;
        }

        label-switched-path to-pe5 {
            to 192.0.2.5;
        }
        interface all;
    }
}
```

Configuring the VPN Interfaces

Step-by-Step Procedure

Router PE2 is one end of the Layer 2 VPN. Router PE3 is performing the Layer 2 VPN stitching between the Layer 2 VPN and the Layer 3 VPN. Router PE3 uses the logical tunnel interface (lt interface) configured with different logical interface units applied under two different Layer 2 VPN instances. The packet is looped though the lt interface configured on Router PE3. The configuration of Router PE5 contains the PE-CE interface.

1. On Router PE2, configure the ge-1/0/2 interface encapsulation. Include the encapsulation statement and specify the `ethernet-ccc` option (`vlan-ccc` encapsulation is also supported) at the `[edit interfaces ge-1/0/2]` hierarchy level. The encapsulation should be the same in a whole Layer 2 VPN domain (Routers PE2 and PE3). Also, configure interface lo0.

```
[edit]
interfaces {
    ge-1/0/2 {
        encapsulation ethernet-ccc;
        unit 0;
    }
    lo0 {
        unit 0 {
            family inet {
                address 192.0.2.2/24;
            }
        }
    }
}
```

2. On Router PE2, configure the routing instance at the `[edit routing-instances]` hierarchy level. Also, configure the Layer 2 VPN protocol at the `[edit routing-instances` *`routing-instances-name`* `protocols]` hierarchy level. Configure the remote site ID as 3. Site ID 3 represents Router PE3 (Hub-PE). The Layer 2 VPN is using LDP as the signaling protocol. Be aware that in the following example, both the routing instance and the protocol are named `l2vpn`.

```
[edit]
routing-instances {
    l2vpn { # routing instance
        instance-type l2vpn;
        interface ge-1/0/2.0;
        route-distinguisher 65000:2;
        vrf-target target:65000:2;
        protocols {
            l2vpn { # protocol
                encapsulation-type ethernet;
                site CE2 {
                    site-identifier 2;
                    interface ge-1/0/2.0 {
                        remote-site-id 3;
                    }
```

```
                }
            }
        }
    }
}
```

3. On Router PE5, configure the Gigabit Ethernet interface for the PE-CE link `ge-2/0/0` and configure the `lo0` interface.

```
[edit interfaces]
ge-2/0/0 {
    unit 0 {
        family inet {
            address 198.51.100.8/24;
        }
    }
}
lo0 {
    unit 0 {
    }
}
```

4. On Router PE5, configure the Layer 3 VPN routing instance (`L3VPN`) at the `[edit routing-instances]` hierarchy level. Also configure BGP at the `[edit routing-instances L3VPN protocols]` hierarchy level.

```
[edit]
routing-instances {
    L3VPN {
        instance-type vrf;
        interface ge-2/0/0.0;
        route-distinguisher 65000:5;
        vrf-target target:65000:2;
        vrf-table-label;
        protocols {
            bgp {
                group ce5 {
                    neighbor 198.51.100.2 {
                        peer-as 200;
                    }
                }
            }
```

```
        }
    }
}
```

5. In an MX Series router, such as Router PE3, you must create the tunnel services interface to be used for tunnel services. To create the tunnel service interface, include the `bandwidth` statement and specify the amount of bandwidth to reserve for tunnel services in gigabits per second at the `[edit chassis fpc` *`slot-number`* `pic` *`slot-number`* `tunnel-services]` hierarchy level.

```
[edit]
chassis {
    dump-on-panic;
    fpc 1 {
        pic 1 {
            tunnel-services {
                bandwidth 1g;
            }
        }
    }
}
```

6. On Router PE3, configure the Gigabit Ethernet interface.

   Include the `address` statement at the `[edit interfaces ge-1/0/1.0 family inet]` hierarchy level and specify `198.51.100.9/24` as the IP address.

```
[edit]
interfaces {
    ge-1/0/1 {
        unit 0 {
            family inet {
                address 198.51.100.9/24;
            }
        }
    }
}
```

7. On Router PE3, configure the `lt-1/1/10.0` logical tunnel interface at the `[edit interfaces lt-1/1/10 unit 0]` hierarchy level. Router PE3 is the router that is *stitching* the Layer 2 VPN to the Layer 3 VPN using the logical tunnel interface. The configuration of the peer unit interfaces is what makes the interconnection.

To configure the interface, include the `encapsulation` statement and specify the `ethernet-ccc` option. Include the `peer-unit` statement and specify the logical interface unit `1` as the peer tunnel interface. Include the `family` statement and specify the `ccc` option.

```
[edit]
interfaces {
    lt-1/1/10 {
        unit 0 {
            encapsulation ethernet-ccc;
            peer-unit 1;
            family ccc;
        }
    }
}
```

8. On Router PE3, configure the `lt-1/1/10.1` logical tunnel interface at the `[edit interfaces lt-1/1/10 unit 1]` hierarchy level.

   To configure the interface, include the `encapsulation` statement and specify the `ethernet` option. Include the `peer-unit` statement and specify the logical interface unit `0` as the peer tunnel interface. Include the `family` statement and specify the `inet` option. Include the `address` statement at the `[edit interfaces lt-1/1/10 unit 0]` hierarchy level and specify `198.51.100.7/24` as the IPv4 address.

```
[edit]
interfaces {
    lt-1/1/10 {
        unit 1 {
            encapsulation ethernet;
            peer-unit 0;
            family inet {
                address 198.51.100.7/24;
            }
        }
    }
}
```

9. On Router PE3, add the `lt` interface unit 1 to the routing instance at the `[edit routing-instances L3VPN]` hierarchy level. Configure the instance type as `vrf` with `lt` peer-unit 1 as a PE-CE interface to terminate the Layer 2 VPN on Router PE2 into the Layer 3 VPN on Router PE3.

```
[edit]
routing-instances {
    L3VPN {
        instance-type vrf;
        interface ge-1/0/1.0;
        interface lt-1/1/10.1;
        route-distinguisher 65000:33;
        vrf-target target:65000:2;
        vrf-table-label;
        protocols {
            bgp {
                export direct;
                group ce3 {
                    neighbor 198.51.100.10 {
                        peer-as 100;
                    }
                }
            }
        }
    }
}
```

10. On Router PE3, add the `lt` interface unit 0 to the routing instance at the `[edit routing-instances protocols l2vpn]` hierarchy level. Also configure the same vrf target for the Layer 2 VPN and Layer 3 VPN routing instances, so that the routes can be leaked between the instances. The example configuration in the previous step shows the vrf target for the `L3VPN` routing instance. The following example shows the vrf target for the `l2vpn` routing instance.

```
[edit]
routing-instances {
    l2vpn {
        instance-type l2vpn;
        interface lt-1/1/10.0;
        route-distinguisher 65000:3;
        vrf-target target:65000:2;
        protocols {
            l2vpn {
```

```
                    encapsulation-type ethernet;
                    site CE3 {
                        site-identifier 3;
                        interface lt-1/1/10.0 {
                            remote-site-id 2;
                        }
                    }
                }
            }
        }
    }
}
```

11. On Router PE3, configure the `policy-statement` statement to export the routes learned from the directly connected `lt` interface unit 1 to all the CE routers for connectivity, if needed.

```
[edit]
policy-options {
    policy-statement direct {
        term 1 {
            from protocol direct;
            then accept;
        }
    }
}
```

Results

The following output shows the full configuration of Router PE2:

Router PE2

```
interfaces {
    xe-0/1/0 {
        unit 0 {
            family inet {
                address 10.10.2.2/30;
            }
            family mpls;
        }
    }
    xe-0/2/0 {
```

```
        unit 0 {
            family inet {
                address 10.10.5.1/30;
            }
            family mpls;
        }
    }
    xe-0/3/0 {
        unit 0 {
            family inet {
                address 10.10.4.1/30;
            }
            family mpls;
        }
    }
    ge-1/0/2 {
        encapsulation ethernet-ccc;
        unit 0;
    }
    fxp0 {
        apply-groups [ re0 re1 ];
    }
    lo0 {
        unit 0 {
            family inet {
                address 192.0.2.2/24;
            }
        }
    }
}
routing-options {
    static {
        route 172.0.0.0/8 next-hop 172.19.59.1;
    }
    autonomous-system 65000;
}
protocols {
    mpls {
        interface all;
        interface fxp0.0 {
            disable;
        }
    }
```

```
    bgp {
        group RR {
            type internal;
            local-address 192.0.2.2;
            family l2vpn {
                signaling;
            }
            neighbor 192.0.2.7;
        }
    }
    ospf {
        traffic-engineering;
        area 0.0.0.0 {
            interface all;
            interface fxp0.0 {
                disable;
            }
        }
    }
    ldp {
        interface all;
        interface fxp0.0 {
            disable;
        }
    }
}
routing-instances {
    l2vpn {
        instance-type l2vpn;
        interface ge-1/0/2.0;
        route-distinguisher 65000:2;
        vrf-target target:65000:2;
        protocols {
            l2vpn {
                encapsulation-type ethernet;
                site CE2 {
                    site-identifier 2;
                    interface ge-1/0/2.0 {
                        remote-site-id 3;
                    }
                }
            }
        }
```

```
    }
}
```

The following output shows the final configuration of Router PE5:

Router PE5

```
interfaces {
    ge-0/0/0 {
        unit 0 {
            family inet {
                address 10.10.4.2/30;
            }
            family mpls;
        }
    }
    xe-0/1/0 {
        unit 0 {
            family inet {
                address 10.10.6.2/30;
            }
            family mpls;
        }
    }
    ge-1/0/0 {
        unit 0 {
            family inet {
                address 10.10.9.1/30;
            }
            family mpls;
        }
    }
    xe-1/1/0 {
        unit 0 {
            family inet {
                address 10.10.3.2/30;
            }
            family mpls;
        }
    }
    ge-2/0/0 {
        unit 0 {
```

```
            family inet {
                address 198.51.100.8/24;
            }
        }
    }
    lo0 {
        unit 0 {
            family inet {
                address 192.0.2.5/24;
            }
        }
    }
}
routing-options {
    static {
        route 172.0.0.0/8 next-hop 172.19.59.1;
    }
    autonomous-system 65000;
}
protocols {
    rsvp {
        interface all {
            link-protection;
        }
        interface fxp0.0 {
            disable;
        }
    }
    mpls {
        label-switched-path to-RR {
            to 192.0.2.7;
        }
        label-switched-path to-PE2 {
            to 192.0.2.2;
        }
        label-switched-path to-PE3 {
            to 192.0.2.3;
        }
        label-switched-path to-PE4 {
            to 192.0.2.4;
        }
        label-switched-path to-PE1 {
            to 192.0.2.1;
```

```
        }
        interface all;
        interface fxp0.0 {
            disable;
        }

    }
    bgp {
        group to-rr {
            type internal;
            local-address 192.0.2.5;
            family inet-vpn {
                unicast;
            }
            family l2vpn {
                signaling;
            }
            neighbor 192.0.2.7;
        }
    }
    ospf {
        traffic-engineering;
        area 0.0.0.0 {
            interface all;
            interface fxp0.0 {
                disable;
            }
        }
    }
    ldp {
        interface all;
        interface fxp0.0 {
            disable;
        }
    }
}
routing-instances {
    L3VPN {
        instance-type vrf;
        interface ge-2/0/0.0;
        route-distinguisher 65000:5;
        vrf-target target:65000:2;
        vrf-table-label;
```

```
        protocols {
            bgp {
                group ce5 {
                    neighbor 198.51.100.2 {
                        peer-as 200;
                    }
                }
            }
        }
    }
}
```

The following output shows the final configuration of Router PE3:

Router PE3

```
chassis {
    dump-on-panic;
    fpc 1 {
        pic 1 {
            tunnel-services {
                bandwidth 1g;
            }
        }
    }
    network-services ip;
}
interfaces {
    ge-1/0/1 {
        unit 0 {
            family inet {
                address 198.51.100.9/24;
            }
        }
    }
    lt-1/1/10 {
        unit 0 {
            encapsulation ethernet-ccc;
            peer-unit 1;
            family ccc;
        }
        unit 1 {
```

```
            encapsulation ethernet;
            peer-unit 0;
            family inet {
                address 198.51.100.7/24;
            }
        }
    }
    xe-2/0/0 {
        unit 0 {
            family inet {
                address 10.10.20.2/30;
            }
            family mpls;
        }
    }
    xe-2/1/0 {
        unit 0 {
            family inet {
                address 10.10.6.1/30;
            }
            family mpls;
        }
    }
    xe-2/2/0 {
        unit 0 {
            family inet {
                address 10.10.5.2/30;
            }
            family mpls;
        }
    }
    xe-2/3/0 {
        unit 0 {
            family inet {
                address 10.10.1.2/30;
            }
            family mpls;
        }
    }
    lo0 {
        unit 0 {
            family inet {
                address 192.0.2.3/24;
```

```
            }
        }
    }
}
routing-options {
    static {
        route 172.0.0.0/8 next-hop 172.19.59.1;
    }
    autonomous-system 65000;
}
protocols {
    rsvp {
        interface all;
        interface fxp0.0 {
            disable;
        }
    }
    mpls {
        label-switched-path to-RR {
            to 192.0.2.7;
        }
        label-switched-path to-PE2 {
            to 192.0.2.2;
        }
        label-switched-path to-PE5 {
            to 192.0.2.5;
        }
        label-switched-path to-PE4 {
            to 192.0.2.4;
        }
        label-switched-path to-PE1 {
            to 192.0.2.1;
        }
        interface all;
        interface fxp0.0 {
            disable;
        }
    }
    bgp {
        group RR {
            type internal;
            local-address 192.0.2.3;
            family inet-vpn {
```

```
                unicast;
            }
            family l2vpn {
                signaling;
            }
            neighbor 192.0.2.7;
        }
    }
    ospf {
        traffic-engineering;
        area 0.0.0.0 {
            interface all;
            interface fxp0.0 {
                disable;
            }
        }
    }
    ldp {
        interface all;
        interface fxp0.0 {
            disable;
        }
    }
}
policy-options {
    policy-statement direct {
        term 1 {
            from protocol direct;
            then accept;
        }
    }
}
routing-instances {
    L3VPN {
        instance-type vrf;
        interface ge-1/0/1.0;
        interface lt-1/1/10.1;
        route-distinguisher 65000:33;
        vrf-target target:65000:2;
        vrf-table-label;
        protocols {
            bgp {
                export direct;
```

```
                group ce3 {
                    neighbor 198.51.100.10 {
                        peer-as 100;
                    }
                }
            }
        }
    }
    l2vpn {
        instance-type l2vpn;
        interface lt-1/1/10.0;
        route-distinguisher 65000:3;
        vrf-target target:65000:2;
        protocols {
            l2vpn {
                encapsulation-type ethernet;
                site CE3 {
                    site-identifier 3;
                    interface lt-1/1/10.0 {
                        remote-site-id 2;
                    }
                }
            }
        }
    }
}
```

Verification

IN THIS SECTION

Verify the Layer 2 VPN-to-Layer 3 VPN interconnection:

Verifying Router PE2 VPN Interface

Purpose

Check that the Layer 2 VPN is up and working at the Router PE2 interface and that all the routes are there.

Action

1. Use the `show l2vpn connections` command to verify that the connection site ID is 3 for Router PE3 and that the status is `Up`.

```
user@PE2> show l2vpn connections
Layer-2 VPN connections:
Legend for connection status (St)
EI -- encapsulation invalid      NC -- interface encapsulation not CCC/TCC/VPLS
EM -- encapsulation mismatch     WE -- interface and instance encaps not same
VC-Dn -- Virtual circuit down    NP -- interface hardware not present
CM -- control-word mismatch      -> -- only outbound connection is up
CN -- circuit not provisioned    <- -- only inbound connection is up
OR -- out of range               Up -- operational
OL -- no outgoing label          Dn -- down
LD -- local site signaled down   CF -- call admission control failure
RD -- remote site signaled down  SC -- local and remote site ID collision
LN -- local site not designated  LM -- local site ID not minimum designated
RN -- remote site not designated RM -- remote site ID not minimum designated
XX -- unknown connection status  IL -- no incoming label
MM -- MTU mismatch               MI -- Mesh-Group ID not available
BK -- Backup connection          ST -- Standby connection
PF -- Profile parse failure      PB -- Profile busy
RS -- remote site standby

Legend for interface status
Up -- operational
Dn -- down

Instance: l2vpn
  Local site: CE2 (2)
    connection-site     Type  St     Time last up        # Up trans
    3                   rmt   Up     Jan  7 14:14:37 2010         1
      Remote PE: 192.0.2.3, Negotiated control-word: Yes (Null)
```

```
        Incoming label: 800000, Outgoing label: 800001
        Local interface: ge-1/0/2.0, Status: Up, Encapsulation: ETHERNET
```

2. Use the `show route table` command to verify that the Layer 2 VPN route is present and that there is a next hop of `10.10.5.2` through the `xe-0/2/0.0` interface. The following output verifies that the Layer 2 VPN routes are present in the l2vpn.l2vpn.0 table. Similar output should be displayed for Router PE3.

```
user@PE2> show route table l2vpn.l2vpn.0
l2vpn.l2vpn.0: 2 destinations, 2 routes (2 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

65000:2:2:3/96
                   *[L2VPN/170/-101] 02:40:35, metric2 1
                      Indirect
65000:3:3:1/96
                   *[BGP/170] 02:40:35, localpref 100, from 192.0.2.7
                      AS path: I
                    > to 10.10.5.2 via xe-0/2/0.0
```

3. Verify that Router PE2 has a Layer 2 VPN MPLS label pointing to the LDP label to Router PE3 in both directions (PUSH and POP).

```
user@PE2> show route table mpls.0
 mpls.0: 13 destinations, 13 routes (13 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

0                  *[MPLS/0] 1w3d 08:57:41, metric 1
                      Receive
1                  *[MPLS/0] 1w3d 08:57:41, metric 1
                      Receive
2                  *[MPLS/0] 1w3d 08:57:41, metric 1
                      Receive
300560             *[LDP/9] 19:45:53, metric 1
                    > to 10.10.2.1 via xe-0/1/0.0, Pop
300560(S=0)        *[LDP/9] 19:45:53, metric 1
                    > to 10.10.2.1 via xe-0/1/0.0, Pop
301008             *[LDP/9] 19:45:53, metric 1
                    > to 10.10.4.2 via xe-0/3/0.0, Swap 299856
301536             *[LDP/9] 19:45:53, metric 1
                    > to 10.10.4.2 via xe-0/3/0.0, Pop
301536(S=0)        *[LDP/9] 19:45:53, metric 1
```

```
                     > to 10.10.4.2 via xe-0/3/0.0, Pop
301712             *[LDP/9] 16:14:52, metric 1
                     > to 10.10.5.2 via xe-0/2/0.0, Swap 315184
301728             *[LDP/9] 16:14:52, metric 1
                     > to 10.10.5.2 via xe-0/2/0.0, Pop
301728(S=0)        *[LDP/9] 16:14:52, metric 1
                     > to 10.10.5.2 via xe-0/2/0.0, Pop
800000             *[L2VPN/7] 02:40:35
                     > via ge-1/0/2.0, Pop       Offset: 4
ge-1/0/2.0         *[L2VPN/7] 02:40:35, metric2 1
                     > to 10.10.5.2 via xe-0/2/0.0, Push 800001 Offset: -4
```

Meaning

The `l2vpn` routing instance is up at interface `ge-1/0/2` and the Layer 2 VPN route is shown in table l2vpn.l2vpn.0. Table `mpls.0` shows the Layer 2 VPN routes used to forward the traffic using an LDP label.

Verifying Router PE3 VPN Interface

Purpose

Check that the Layer 2 VPN connection from Router PE2 and Router PE3 is `Up` and working.

Action

1. Verify that the BGP session with the route reflector for the family `l2vpn-signaling` and the family `inet-vpn` is established.

```
user@PE3> show bgp summary
Groups: 2 Peers: 2 Down peers: 0
Table        Tot Paths  Act Paths Suppressed    History Damp State    Pending
bgp.l2vpn.0          1          1          0          0          0          0
bgp.L3VPN.0          1          1          0          0          0          0
Peer         AS    InPkt    OutPkt   OutQ   Flaps Last Up/Dwn   State|#Active /Received/
Accepted/Damped...
192.0.2.7  65000    2063      2084      0       1    15:35:16   Establ
  bgp.l2vpn.0: 1/1/1/0
  bgp.L3VPN.0: 1/1/1/0
  L3VPN.inet.0: 1/1/1/0
  l2vpn.l2vpn.0: 1/1/1/0
```

2. The following output verifies the Layer 2 VPN route and the label associated with it.

```
user@PE3> show route table l2vpn.l2vpn.0 detail
l2vpn.l2vpn.0: 2 destinations, 2 routes (2 active, 0 holddown, 0 hidden)
65000:2:2:3/96 (1 entry, 1 announced)
        *BGP    Preference: 170/-101
                Route Distinguisher: 65000:2
                Next hop type: Indirect
                Next-hop reference count: 4
                Source: 192.0.2.7
                Protocol next hop: 192.0.2.2
                Indirect next hop: 2 no-forward
                State: <Secondary Active Int Ext>
                Local AS: 65000 Peer AS: 65000
                Age: 2:45:52    Metric2: 1
                Task: BGP_65000.192.0.2.7+60585
                Announcement bits (1): 0-l2vpn-l2vpn
                AS path: I (Originator) Cluster list:  192.0.2.7
                AS path:  Originator ID: 192.0.2.2
                Communities: target:65000:2 Layer2-info: encaps:ETHERNET, control
flags:Control-Word, mtu: 0, site preference: 100  Accepted
                Label-base: 800000, range: 2, status-vector: 0x0
                Localpref: 100
                Router ID: 192.0.2.7
                Primary Routing Table bgp.l2vpn.0
```

3. The following output show the L2VPN MPLS.0 route in the mpls.0 route table.

```
user@PE3> show route table mpls.0
mpls.0: 21 destinations, 21 routes (21 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

0                  *[MPLS/0] 1w3d 09:05:41, metric 1
                      Receive
1                  *[MPLS/0] 1w3d 09:05:41, metric 1
                      Receive
2                  *[MPLS/0] 1w3d 09:05:41, metric 1
                      Receive
16                 *[VPN/0] 15:59:24
                      to table L3VPN.inet.0, Pop
315184             *[LDP/9] 16:21:53, metric 1
```

```
                       > to 10.10.20.1 via xe-2/0/0.0, Pop
315184(S=0)           *[LDP/9] 16:21:53, metric 1
                       > to 10.10.20.1 via xe-2/0/0.0, Pop
315200                *[LDP/9] 01:13:44, metric 1
                         to 10.10.20.1 via xe-2/0/0.0, Swap 625297
                       > to 10.10.6.2 via xe-2/1/0.0, Swap 299856
315216                *[LDP/9] 16:21:53, metric 1
                       > to 10.10.6.2 via xe-2/1/0.0, Pop
315216(S=0)           *[LDP/9] 16:21:53, metric 1
                       > to 10.10.6.2 via xe-2/1/0.0, Pop
315232                *[LDP/9] 16:21:45, metric 1
                       > to 10.10.1.1 via xe-2/3/0.0, Pop
315232(S=0)           *[LDP/9] 16:21:45, metric 1
                       > to 10.10.1.1 via xe-2/3/0.0, Pop
315248                *[LDP/9] 16:21:53, metric 1
                       > to 10.10.5.1 via xe-2/2/0.0, Pop
315248(S=0)           *[LDP/9] 16:21:53, metric 1
                       > to 10.10.5.1 via xe-2/2/0.0, Pop
315312                *[RSVP/7] 15:02:40, metric 1
                       > to 10.10.6.2 via xe-2/1/0.0, label-switched-path to-pe5
315312(S=0)           *[RSVP/7] 15:02:40, metric 1
                       > to 10.10.6.2 via xe-2/1/0.0, label-switched-path to-pe5
315328                *[RSVP/7] 15:02:40, metric 1
                       > to 10.10.20.1 via xe-2/0/0.0, label-switched-path to-RR
315360                *[RSVP/7] 15:02:40, metric 1
                       > to 10.10.20.1 via xe-2/0/0.0, label-switched-path to-RR
316272                *[RSVP/7] 01:13:27, metric 1
                       > to 10.10.6.2 via xe-2/1/0.0, label-switched-path Bypass->10.10.9.1
316272(S=0)           *[RSVP/7] 01:13:27, metric 1
                       > to 10.10.6.2 via xe-2/1/0.0, label-switched-path Bypass->10.10.9.1
800001                *[L2VPN/7] 02:47:33
                       > via lt-1/1/10.0, Pop       Offset: 4
lt-1/1/10.0           *[L2VPN/7] 02:47:33, metric2 1
                       > to 10.10.5.1 via xe-2/2/0.0, Push 800000 Offset: -4
```

4. Use the `show route table mpls.0` command with the `detail` option to see the BGP attributes of the route such as next-hop type and label operations.

```
user@PE5> show route table mpls.0 detail
lt-1/1/10.0 (1 entry, 1 announced)
        *L2VPN  Preference: 7
                Next hop type: Indirect
```

```
                  Next-hop reference count: 2
                  Next hop type: Router, Next hop index: 607
                  Next hop: 10.10.5.1 via xe-2/2/0.0, selected
                  Label operation: Push 800000 Offset: -4
                  Protocol next hop: 192.0.2.2
                  Push 800000 Offset: -4
                  Indirect next hop: 8cae0a0 1048574
                  State:  <Active Int>
                  Age: 2:46:34    Metric2: 1
                  Task: Common L2 VC
                  Announcement bits (2): 0-KRT 2-Common L2 VC
                  AS path: I
                  Communities: target:65000:2 Layer2-info: encaps:ETHERNET, control
flags:Control-Word, mtu: 0, site preference: 100
```

Verifying End-to-End connectivity from Router CE2 to Router CE5 and Router CE3

Purpose

Check the connectivity between Routers CE2, CE3, and CE5.

Action

1. Ping the Router CE3 IP address from Router CE2.

```
user@CE2> ping 198.51.100.10 # CE3 IP address
PING 198.51.100.10 (198.51.100.10): 56 data bytes
64 bytes from 198.51.100.10: icmp_seq=0 ttl=63 time=0.708 ms
64 bytes from 198.51.100.10: icmp_seq=1 ttl=63 time=0.610 ms
```

2. Ping the Router CE5 IP address from Router CE2.

```
user@CE2> ping 198.51.100.2 # CE5 IP address
PING 198.51.100.2 (198.51.100.2): 56 data bytes
64 bytes from 198.51.100.2: icmp_seq=0 ttl=62 time=0.995 ms
64 bytes from 198.51.100.2: icmp_seq=1 ttl=62 time=1.005 ms
```

RELATED DOCUMENTATION

Understanding Layer 2 VPNs

Understanding Layer 3 VPNs | **6**

CHAPTER 3

Interprovider and Carrier-of-Carrier VPNs

IN THIS CHAPTER

Interprovider and Carrier-of-Carriers VPNs

IN THIS SECTION

Traditional VPNs, Interprovider VPNs, and Carrier-of-Carriers VPNs

As VPNs are deployed on the Internet, the customer of a VPN service provider might be another service provider rather than an end customer. The customer service provider depends on the VPN service provider to deliver a VPN transport service between the customer service provider's points of presence (POPs) or regional networks.

If the customer service provider's sites have different autonomous system (AS) numbers, then the VPN transit service provider supports carrier-of-carrier VPN service for the interprovider VPN service. If the customer service provider's sites have the same AS number, then the VPN transit service provider delivers a carrier-of-carriers VPN service.

There are several different methods for enabling interprovider VPNs based on RFC 4364, `BGP/MPLS IP Virtual Private Networks (VPNs)`:

- Interprovider Layer 3 VPN Option A—Interprovider VRF-to-VRF connections at the AS boundary routers (ASBR) (not very scalable).
- Interprovider Layer 3 VPN Option B—Interprovider EBGP redistribution of labeled VPN-IPv4 routes from AS to neighboring AS (somewhat scalable).
- Interprovider Layer 3 VPN Option C—Interprovider multihop EBGP redistribution of labeled VPN-IPv4 routes between source and destination ASs, with EBGP redistribution of labeled IPv4 routes from AS to neighboring AS (very scalable).

In traditional IP routing architectures, there is a clear distinction between internal routes and external routes. From the perspective of an Internet service provider (ISP), internal routes include all the provider's internal links (including BGP next hops) and loopback interfaces. These internal routes are exchanged with other routing platforms in the ISP's network by means of an interior gateway protocol (IGP), such as OSPF or IS-IS. All routes learned at Internet peering points or from customer sites are classified as external routes and are distributed by means of an exterior gateway protocol (EGP) such as BGP. In traditional IP routing architectures, the number of internal routes is typically much smaller than the number of external routes.

Understanding Interprovider and Carrier-of-Carriers VPNs

All interprovider and carrier-of-carriers VPNs share the following characteristics:

- Each interprovider or carrier-of-carriers VPN customer must distinguish between internal and external customer routes.
- Internal customer routes must be maintained by the VPN service provider in its PE routers.
- External customer routes are carried only by the customer's routing platforms, not by the VPN service provider's routing platforms.

The key difference between interprovider and carrier-of-carriers VPNs is whether the customer sites belong to the same AS or to separate ASs:

- "Interprovider VPNs" on page 447—The customer sites belong to different ASs. You need to configure EBGP to exchange the customer's external routes.
- "Understanding Carrier-of-Carriers VPNs" on page 529—The customer sites belong to the same AS. You need to configure IBGP to exchange the customer's external routes.

In general, each service provider in a VPN hierarchy is required to maintain its own internal routes in its P routers, and the internal routes of its customers in its PE routers. By recursively applying this rule, it is possible to create a hierarchy of VPNs.

The following are definitions of the types of PE routers specific to interprovider and carrier-of-carriers VPNs:

- The AS border router is located at the AS border and handles traffic leaving and entering the AS.
- The end PE router is the PE router in the customer VPN; it is connected to the CE router at the end customer's site.

Interprovider and Carrier-of-Carrier VPNs Example Terminology

IN THIS SECTION

`bgp.l3vpn.0`

The table on the provider edge (PE) router in which the VPN-IPv4 routes that are received from another PE router are stored. Incoming routes are checked against the `vrf-import` statements from all the VPNs configured on the PE router. If there is a match, the VPN–Internet Protocol version 4 (IPv4) route is added to the bgp.l3vpn.0 table. To view the bgp.l3vpn.0 table, issue the `show route table bgp.l3vpn.0` command.

routing-instance-name. inet.0

- The routing table for a specific routing instance. For example, a routing instance called VPN-A has a routing table called VPN-A.inet.0. Routes are added to this table in the following ways:
  - They are sent from a customer edge (CE) router configured within the VPN-A routing instance.
  - They are advertised from a remote PE router that passes the `vrf-import` policy configured within VPN-A (to view the route, run the `show route` command). IPv4 (not VPN-IPv4) routes are stored in this table.

`vrf-import` *`policy-name`*

An import policy configured on a particular routing instance on a PE router. This policy is required for the configuration of interprovider and carrier-of-carriers VPNs. It is applied to VPN-IPv4 routes learned from another PE router or a route reflector.

`vrf-export` *`policy-name`*

An export policy configured on a particular routing instance on a PE router. It is required for the configuration of interprovider and carrier-of-carriers VPNs. It is applied to VPN-IPv4 routes (originally learned from locally connected CE routers as IPv4 routes), which are advertised to another PE router or route reflector.

`MP-EBGP`

The multiprotocol external BGP (MP-EBGP) mechanism is used to export VPN-IPv4 routes across an autonomous system (AS) boundary. To apply this mechanism, use the `labeled-unicast` statement at the `[edit protocols bgp group` *`group-name`* `family inet]` hierarchy level.

Supported Carrier-of-Carriers and Interprovider VPN Standards

Junos OS substantially supports the following RFCs, which define standards for carrier-of-carriers and interprovider virtual private networks (VPNs).

- RFC 3107, *Carrying Label Information in BGP-4*
- RFC 3916, *Requirements for Pseudo-Wire Emulation Edge-to-Edge (PWE3)*

  Supported on MX Series routers with the Channelized OC3/STM1 (Multi-Rate) Circuit Emulation MIC with SFP.
- RFC 3985, *Pseudo Wire Emulation Edge-to-Edge (PWE3) Architecture*

  Supported on MX Series routers with the Channelized OC3/STM1 (Multi-Rate) Circuit Emulation MIC with SFP.
- RFC 4364, *BGP/MPLS IP Virtual Private Networks (VPNs)*
- RFC 6368, *Internal BGP as the Provider/Customer Edge Protocol for BGP/MPLS IP Virtual Private Networks (VPNs)*

SEE ALSO

Supported VPWS Standards

Supported Layer 2 VPN Standards

Supported Layer 3 VPN Standards | **10**

Supported Multicast VPN Standards | **583**

Supported VPLS Standards

Supported Standards for BGP

Accessing Standards Documents on the Internet

RELATED DOCUMENTATION

Understanding Carrier-of-Carriers VPNs | 529

MPLS Feature Support on QFX Series and EX4600 Switches

Interprovider VPNs

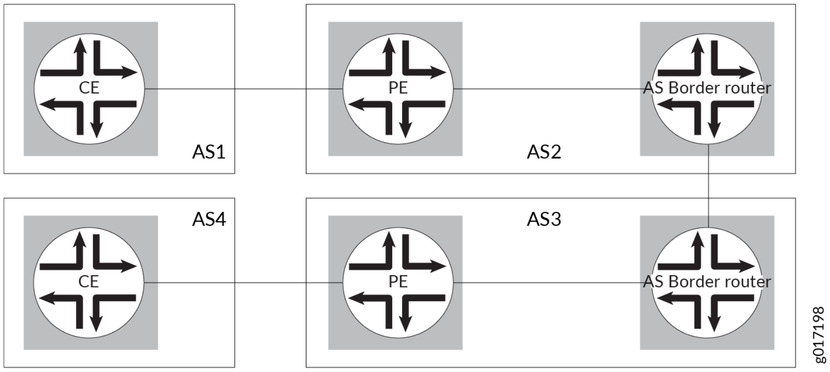
IN THIS SECTION

Interprovider VPNs

IN THIS SECTION

Interprovider VPNs provide connectivity between separate ASs. This functionality might be used by a VPN customer who has connections to several different service providers, or different connections to the same service provider in different geographic regions, each of which has a different AS. Figure 46 on page 448 illustrates the type of network topology used by an interprovider VPN.

Figure 46: Interprovider VPN Network Topology



The following sections describe the ways you can configure an interprovider VPN:

Linking VRF Tables Between Autonomous Systems

You can connect two separate ASs by simply linking the VPN routing and forwarding (VRF) table in the AS border router (ASBR) of one AS to the VRF table in the ASBR in the other AS. Each ASBR must include a VRF routing instance for each VPN configured in both service provider networks. You then configure an IP session between the two ASBRs. In effect, the ASBRs treat each other as customer edge (CE) routers.

Because of the complexity of the configuration, particularly with regard to scaling, this method is not recommended. The details of this configuration are not provided with documentation.

Configuring Next Generation Layer 3 VPNs Options A, B, and C

For next generation Layer 3 VPNs, the PE routers within an AS use multiprotocol external BGP (MP-EBGP) to distribute labeled VPN–Internet Protocol version 4 (IPv4) routes to an ASBR or to a route reflector of which the ASBR is a client. The ASBR uses multiprotocol external BGP (MP-EBGP) to distribute the labeled VPN-IPv4 routes to its peer ASBR in the neighboring AS. The peer ASBR then uses MP-IBGP to distribute labeled VPN-IPv4 routes to PE routers, or to a route reflector of which the PE routers are a client.

You can configure both unicast (Junos OS Release 9.5 and later) and multicast (Junos OS Release 12.1 and later) next generation Layer 3 VPNs across ASs. The Junos OS software supports next generation Layer 3 VPNs option A, option B, and option C:

- Option A—This is simple though less scaleable interprovider VPN solution to the problem of providing VPN services to a customer that has different sites, not all of which can use the same service provider. In this implementation, the VPN routing and forwarding (VRF) table in the ASBR of one AS is linked to the VRF table in the ASBR in the other AS. Each ASBR must include a VRF instance for each VPN configured in both service provider networks. Then an IGP or BGP must be configured between the ASBRs.

- Option B—For this interprovider VPN solution, the customer requires VPN services for different sites, yet the same service provider is not available for all of those sites. With option B, the ASBR routers keep all VPN-IPv4 routes in the routing information base (RIB), and the labels associated with the prefixes are kept in the forwarding information base (FIB). Because the RIB and FIB tables can take too much of the respective allocated memory, this solution is not very scalable for an interprovider VPN. If a transit service provider is used between service provider 1 and service provider 2, the transit service provider also has to keep all VPN-IPv4 routes in the RIB and the corresponding labels in the FIB. The ASBRs at the transit service provider have the same functionality as ASBRs at service provider 1 or service provider 2 in this solution. The PE routers within each AS use multiprotocol internal BGP (MP-IBGP) to distribute labeled VPN-IPv4 routes to an ASBR or to a route reflector of which the ASBR is a client. The ASBR uses MP-EBGP to distribute the labeled VPN-IPv4 routes to its peer ASBR router in the neighboring AS. The peer ASBR then uses MP-IBGP to distribute labeled VPN-IPv4 routes to PE routers, or to a route reflector of which the PE routers are a client.

- Option C—For this interprovider VPN solution, the customer service provider depends on the VPN service provider to deliver a VPN transport service between the customer service provider's points of presence (POPs) or regional networks. This functionality might be used by a VPN customer who has connections to several different service providers, or different connections to the same service provider in different geographic regions, each of which has a different AS number. For option C, only routes internal to the service provider networks are announced between ASBRs. This is achieved by using the `family inet labeled-unicast` statements in the IBGP and EBGP configuration on the PE routers. Labeled IPv4 (not VPN-IPv4) routes are exchanged by the ASBRs to support MPLS. An MP-EBGP session between the end PE routers is used for the announcement of VPN-IPv4 routes. In this manner, VPN connectivity is provided while keeping VPN-IPv4 routes out of the core network.

Configuring Multihop MP-EBGP Between AS Border Routers

In this type of interprovider VPN configuration, P routers do not need to store all the routes in all the VPNs. Only the PE routers must have all the VPN routes. The P routers simply forward traffic to the PE routers—they do not store or process any information about the packets' destination. The connections between the AS border routers in separate ASs forward traffic between the ASs, much as a label-switched path (LSP) works.

The following are the basic steps you take to configure an interprovider VPN in this manner:

1. Configure multihop EBGP redistribution of labeled VPN-IPv4 routes between the source and destination ASs.

2. Configure EBGP to redistribute labeled IPv4 routes from its AS to neighboring ASs.

3. Configure MPLS on the end PE routers of the VPNs.

SEE ALSO

Example: Configuring Interprovider Layer 3 VPN Option A | 450

Example: Configuring Interprovider Layer 3 VPN Option B | 479

Example: Configuring Interprovider Layer 3 VPN Option C | 498

MBGP Multicast VPN Sites | 597

MBGP Multicast VPN Sites | 597

Example: Configuring Interprovider Layer 3 VPN Option A

IN THIS SECTION

Interprovider Layer 3 VPN Option A provides interprovider VRF-to-VRF connections at the AS boundary routers (ASBRs). Compared to Option B and Option C, Option A is the least scalable solution.

This example provides a step-by-step procedure to configure interprovider Layer 3 VPN option A, which is one of the recommended implementations of MPLS VPN when that service is required by a customer that has more than one AS and but not all of the customer’s ASs can be serviced by the same service provider. It is organized in the following sections:

Requirements

This example uses the following hardware and software components:

- Junos OS Release 9.5 or later.
- Eight M Series, T Series, TX Series, or MX Series Juniper Networks routers.

Overview and Topology

IN THIS SECTION

This is the simplest and least scalable interprovider VPN solution to the problem of providing VPN services to a customer that has different sites, not all of which can use the same service provider (SP).

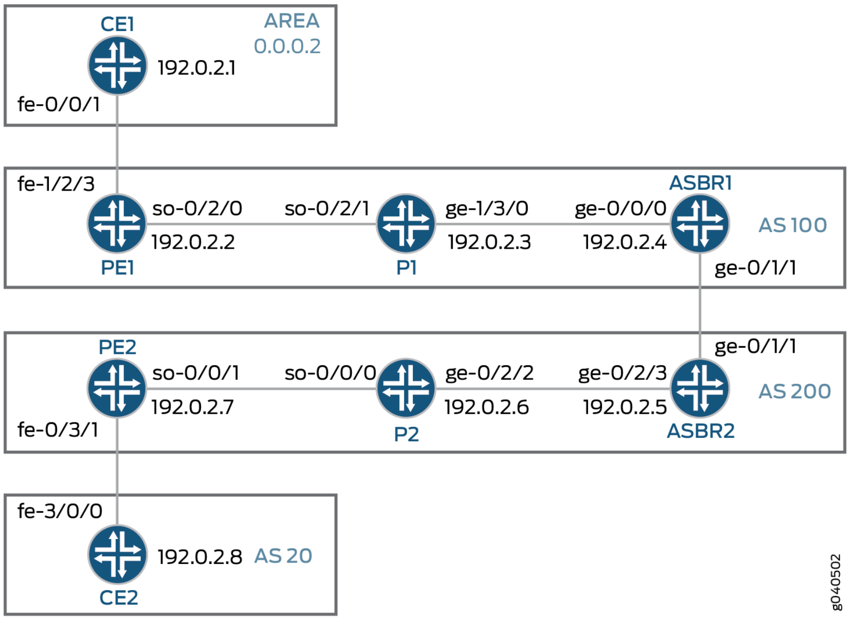
RFC 4364, section 10, refers to this method as Interprovider VRF-to-VRF connections at the AS border routers.

In this configuration:

- The virtual routing and forwarding (*VRF*) table in the ASBR of one AS is linked to the VRF table in the ASBR in the other AS. Each ASBR must contain a VRF instance for every VPN configured in both service provider networks. Then an IGP or BGP must be configured between the ASBRs. This has the disadvantage of limiting scalability.
- In this configuration, the autonomous system boundary routers (ASBRs) at both SPs are configured as regular PE routers, and provide MPLS L3 VPN service to the neighbor SP.
- Each PE router treats the other as if it were a customer edge (CE) router. ASBRs play the role of regular CE routers for the ASBR of the remote SP. ASBRs see each other as CE devices.
- A provider edge (PE) router in one autonomous system (AS) attaches directly to a PE router in another AS.
- The two PE routers are attached by multiple sub-interfaces, at least one for each of the VPNs whose routes need to be passed from AS to AS.
- The PE routers associate each sub-interface with a VPN routing and forwarding (VRF) table, and use EBGP to distribute unlabeled IPv4 addresses to each other.
- In this solution, all common VPNs defined at both PEs must also be defined at one or more ASBRs between the two SPs. This is not a very scalable methodology, especially when a transit SP is used by two regional SPs for interconnection.
- This is a procedure that is simple to configure and it does not require MPLS at the border between ASs. Additionally, it does not scale as well as other recommended procedures.

The topology of the network is shown in Figure 47 on page 452.

Figure 47: Physical Topology of Interprovider Layer 3 VPN Option A



Topology

Configuration

IN THIS SECTION

NOTE: The procedure presented here is written with the assumption that the reader is already familiar with MPLS MVPN configuration. This example focuses on explaining the unique configuration required for carrier-of-carriers solutions for VPN services to different sites.

To configure interprovider layer 3 VPN option A, perform the following tasks:

Configuring Router CE1

Step-by-Step Procedure

1. On Router CE1, configure the IP address and protocol family on the Fast Ethernet interface for the link between Router CE1 and Router PE1. Specify the `inet` address family type.

```
[edit interfaces fe-0/0/1.0]
family inet {
    address 198.51.100.1/24;
}
```

2. On Router CE1, configure the IP address and protocol family on the loopback interface. Specify the `inet` address family type.

```
[edit interfaces lo0]
unit 0 {
    family inet {
        address 192.0.2.1/32;
    }
}
```

3. On Router CE1, configure a routing protocol. The routing protocol can be a static route, RIP, OSPF, ISIS, or EBGP. In this example we configure OSPF. Include the Fast Ethernet interface for the link between Router CE1 and Router PE1 and the logical loopback interface of Router CE1.

```
[edit protocols]
ospf {
    area 0.0.0.2 {
        interface fe-0/0/1.0;
        interface lo0.0;
```

```
    }
}
```

Configuring Router PE1

Step-by-Step Procedure

1. On Router PE1, configure IPv4 addresses on the SONET, Fast Ethernet, and logical loopback interfaces. Specify the `inet` address family on all of the interfaces. Specify the `mpls` address family on the SONET and Fast Ethernet interfaces.

```
[edit interfaces]
so-0/2/0 {
    unit 0 {
        family inet {
            address 192.168.1.9/24;
        }
        family mpls;
    }
}
fe-1/2/3 {
    unit 0 {
        family inet {
            address 198.51.100.2/24;
        }
        family mpls;
    }
}
lo0 {
    unit 0 {
        family inet {
            address 192.0.2.2/32;
        }

    }
}
```

2. On Router PE1, configure the routing instance for VPN2. Specify the `vrf` instance type and specify the customer-facing Fast Ethernet interface. Configure a route distinguisher to create a unique VPN-IPv4 address prefix. Apply the VRF import and export policies to enable the sending and receiving of

route targets. Configure the OSPF protocol within the VRF. Specify the customer-facing Fast Ethernet interface and specify the export policy to export BGP routes into OSPF.

```
[edit routing-instances]
vpn2CE1 {
    instance-type vrf;
    interface fe-1/2/3.0;
    route-distinguisher 1:100;
    vrf-import vpnimport;
    vrf-export vpnexport;
    protocols {
        ospf {
            export bgp-to-ospf;
            area 0.0.0.2 {
                interface fe-1/2/3.0;
            }
        }
    }
}
```

3. On Router PE1, configure the RSVP and MPLS protocols to support the label-switched path (LSP). Configure the LSP to Router ASBR1 and specify the IP address of the logical loopback interface on Router ASBR1. Configure a BGP group. Specify the group type as `internal`. Specify the local address as the logical loopback interface on Router PE1. Specify the neighbor address as the logical loopback interface on Router ASBR1. Specify the `inet-vpn` address family and `unicast` traffic type to enable BGP to carry IPv4 network layer reachability information (NLRI) for VPN routes. Configure the OSPF protocol. Specify the core-facing SONET interface and specify the logical loopback interface on Router PE1.

```
[edit protocols]
rsvp {
    interface so-0/2/0.0;
    interface lo0.0;
}
mpls {
    label-switched-path To-ASBR1 {
        to 192.0.2.4;
    }
    interface so-0/2/0.0;
    interface lo0.0;
}
```

```
bgp {
    group To_ASBR1 {
        type internal;
        local-address 192.0.2.2;
        neighbor 192.0.2.4 {
            family inet-vpn {
                unicast;
            }
        }
    }
}
ospf {
    traffic-engineering;
    area 0.0.0.0 {
        interface so-0/2/0.0;
        interface lo0.0;
    }
}
```

4. On Router PE1, configure the BGP local autonomous system number.

```
[edit routing-options]
autonomous-system 100;
```

5. On Router PE1, configure a policy to export the BGP routes into OSPF.

```
[edit policy-options]
policy-statement bgp-to-ospf {
    term 1 {
        from protocol bgp;
        then accept;
    }
    term 2 {
        then reject;
    }
}
```

6. On Router PE1, configure a policy to add the VRF route target to the routes being advertised for this VPN.

```
[edit policy-options]
policy-statement vpnexport {
    term 1 {
        from protocol ospf;
        then {
            community add test_comm;
            accept;
        }
    }
    term 2 {
        then reject;
    }
}
```

7. On Router PE1, configure a policy to import routes from BGP that have the `test_comm` community attached.

```
[edit policy-options]
policy-statement vpnimport {
    term 1 {
        from {
            protocol bgp;
            community test_comm;
        }
        then accept;
    }
    term 2 {
        then reject;
    }
}
```

8. On Router PE1, define the `test_comm` BGP community with a route target.

```
[edit policy-options]
community test_comm members target:1:100;
```

Configuring Router P1

Step-by-Step Procedure

1. On Router P1, configure IP addresses for the SONET and Gigabit Ethernet interfaces. Enable the interfaces to process the `inet` and `mpls` address families. Configure the IP address for the `lo0.0` loopback interface and enable the interface to process the `inet` address family.

```
[edit interfaces]
so-0/2/1 {
    unit 0 {
        family inet {
            address 192.168.1.4/24;
        }
        family mpls;
    }
}
ge-1/3/0 {
    unit 0 {
        family inet {
            address 192.168.2.5/24;
        }
        family mpls;
    }
}
lo0 {
    unit 0 {
        family inet {
            address 192.0.2.3/32;
        }
    }
}
```

2. On Router P1, configure the RSVP and MPLS protocols to support the LSP. Specify the SONET and Gigabit Ethernet interfaces.

   Configure the OSPF protocol. Specify the SONET and Gigabit Ethernet interfaces and specify the logical loopback interface. Enable OSPF to support traffic engineering extensions.

```
[edit protocols]
rsvp {
```

```
    interface so-0/2/1.0;
    interface ge-1/3/0.0;
    interface lo0.0;
}
mpls {
    interface lo0.0;
    interface ge-1/3/0.0;
    interface so-0/2/1.0;
}
ospf {
    traffic-engineering;
    area 0.0.0.0 {
        interface ge-1/3/0.0;
        interface so-0/2/1.0;
        interface lo0.0;
    }
}
```

Configuring Router ASBR1

Step-by-Step Procedure

1. On Router ASBR1, configure IP addresses for the Gigabit Ethernet interfaces. Enable the interfaces to process the `inet` and `mpls` addresses families. Configure the IP addresses for the `lo0.0` loopback interface and enable the interface to process the `inet` address family.

```
[edit interfaces]
ge-0/0/0 {
    unit 0 {
        family inet {
            address 192.168.2.6/24;
        }
        family mpls;
    }
}
ge-0/1/1 {
    unit 0 {
        family inet {
            address 192.168.3.7/24;
        }
        family mpls;
```

```
    }
}
lo0 {
    unit 0 {
        family inet {
            address 192.0.2.4/32;
        }
    }
}
```

2. On Router ASBR1, configure the `To_ASBR2` routing instance. Specify the `vrf` instance type and specify the core-facing Gigabit Ethernet interface. Configure a route distinguisher to create a unique VPN-IPv4 address prefix. Configure a route target for the VPN. Configure the BGP peer group within the VRF. Specify AS 200 as the peer AS and specify the IP address of the Gigabit Ethernet interface on Router ASBR2 as the neighbor address.

```
[edit routing instances]
To_ASBR2{
    instance-type vrf;
    interface ge-0/1/1.0;
    route-distinguisher 1:100;
    vrf-target target:1:100;
    protocols {
        bgp {
            group To_ASBR2 {
                type external;
                neighbor 192.168.3.8 {
                    peer-as 200;
                }
            }
        }
    }
}
```

3. On Router ASBR1, configure the RSVP and MPLS protocols to support the LSP by specifying the Gigabit Ethernet interface that is facing the P1 router.

Configure the OSPF protocol by specifying the Gigabit Ethernet interface that is facing the P1 router and the logical loopback interface. Enable OSPF to support traffic engineering extensions.

```
[edit protocols]
rsvp {
    interface ge-0/0/0.0;
    interface lo0.0;
}
mpls {
    label-switched-path To_PE1 {
        to 192.0.2.2;
    }
    interface lo0.0;
    interface ge-0/0/0.0;
}
ospf {
    traffic-engineering;
    area 0.0.0.0 {
        interface ge-0/0/0.0;
        interface lo0.0;
    }
}
```

4. On Router ASBR1, create the `To-PE1` internal BGP peer group. Specify the local IP peer address as the local `lo0.0` address. Specify the neighbor IP peer address as the `lo0.0` interface address of Router PE1.

```
[edit protocols]
bgp {
    group To-PE1 {
        type internal;
        local-address 192.0.2.4;
        neighbor 192.0.2.2 {
            family inet-vpn {
                unicast;
            }
        }
    }
}
```

5. On Router ASBR1, configure the BGP local autonomous system number.

```
[edit routing-options]
autonomous-system 100;
```

Configuring Router ASBR2

Step-by-Step Procedure

1. On Router ASBR2, configure IP addresses for the Gigabit Ethernet interfaces. Enable the interfaces to process the `inet` and `mpls` address families. Configure the IP address for the `lo0.0` loopback interface and enable the interface to process the `inet` address family.

```
[edit interfaces]
ge-0/1/1 {
    unit 0 {
        family inet {
            address 192.168.3.8/24;
        }
        family mpls;
    }
}
ge-0/2/3 {
    unit 0 {
        family inet {
            address 192.168.4.10/24;
        }
        family mpls;
    }
}
lo0 {
    unit 0 {
        family inet {
            address 192.0.2.5/32;
        }
    }
}
```

2. On Router ASBR2, configure the `To_ASBR1` routing instance. Specify the `vrf` instance type and specify the core-facing Gigabit Ethernet interface. Configure a route distinguisher to create a unique VPN-

IPv4 address prefix. Configure a route target for the VPN. Configure the BGP peer group within the VRF. Specify AS 100 as the peer AS and specify the IP address of the Gigabit Ethernet interface on Router ASBR1 as the neighbor address.

```
[edit routing-instances]
To_ASBR1 {
    instance-type vrf;
    interface ge-0/1/1.0;
    route-distinguisher 1:100;
    vrf-target target:1:100;
    protocols {
        bgp {
            group To_ASBR1 {
                type external;
                neighbor 192.168.3.7 {
                    peer-as 100;
                }
            }
        }
    }
}
```

3. On Router ASBR2, configure the RSVP and MPLS protocols to support the LSP by specifying the Gigabit Ethernet interface that is facing the P2 router.

   Configure the OSPF protocol by specifying the Gigabit Ethernet interface that is facing the P2 router and the logical loopback interface. Enable OSPF to support traffic engineering extensions.

```
[edit protocols]
rsvp {
    interface ge-0/2/3.0;
    interface lo0.0;
}
mpls {
    label-switched-path To_PE2 {
        to 192.0.2.7;
    }
    interface lo0.0;
    interface ge-0/2/3.0;
}
ospf {
```

```
    traffic-engineering;
    area 0.0.0.0 {
        interface ge-0/2/3.0;
        interface lo0.0;
    }
}
```

4. On Router ASBR2, create the `To-PE2` internal BGP peer group. Specify the local IP peer address as the local `lo0.0` address. Specify the neighbor IP peer address as the `lo0.0` interface address of Router PE2.

```
[edit protocols]
bgp {
    group To-PE2 {
        type internal;
        local-address 192.0.2.5;
        neighbor 192.0.2.7 {
            family inet-vpn {
                unicast;
            }
        }
    }
```

5. On Router ASBR2, configure the BGP local autonomous system number.

```
[edit routing-options]
autonomous-system 200;
```

Configuring Router P2

Step-by-Step Procedure

1. On Router P2, configure IP addresses for the SONET and Gigabit Ethernet interfaces. Enable the interfaces to process the `inet` and `mpls` address families. Configure the IP address for the `lo0.0` loopback interface and enable the interface to process the `inet` address family.

```
[edit interfaces]
so-0/0/0 {
    unit 0 {
        family inet {
```

```
            address 192.168.5.11/24;
        }
        family mpls;
    }
}
ge-0/2/2 {
    unit 0 {
        family inet {
            address 192.168.4.12/24;
        }
        family mpls;
    }
}
lo0 {
    unit 0 {
        family inet {
            address 192.0.2.6/32;
        }
    }
}
```

2. On Router P2, configure the RSVP and MPLS protocols to support the LSP. Specify the SONET and Gigabit Ethernet interfaces.

   Configure the OSPF protocol. Specify the SONET and Gigabit Ethernet interfaces and specify the logical loopback interface. Enable OSPF to support traffic engineering extensions.

```
[edit protocols]
rsvp {
    interface so-0/0/0.0;
    interface ge-0/2/2.0;
    interface lo0.0;
}
mpls {
    interface lo0.0;
    interface ge-0/2/2.0;
    interface so-0/0/0.0;
}
ospf {
    traffic-engineering;
    area 0.0.0.0 {
        interface ge-0/2/2.0;
```

```
        interface so-0/0/0.0;
        interface lo0.0;
    }
}
```

Configuring Router PE2

Step-by-Step Procedure

1. On Router PE2, configure IPv4 addresses on the SONET, Fast Ethernet, and logical loopback interfaces. Specify the `inet` address family on all of the interfaces. Specify the `mpls` address family on the SONET and Fast Ethernet interfaces.

```
[edit interfaces]
so-0/0/1 {
    unit 0 {
        family inet {
            address 192.168.5.12/24;
        }
        family mpls;
    }
}
fe-0/3/1 {
    unit 0 {
        family inet {
            address 192.168.6.13/24;
        }
        family mpls;
    }
lo0 {
    unit 0 {
        family inet {
            address 192.0.2.7/32;
        }
    }
}
```

2. On Router PE2, configure the routing instance for VPN2. Specify the `vrf` instance type and specify the customer-facing Fast Ethernet interface. Configure a route distinguisher to create a unique VPN-IPv4 address prefix. Apply the VRF import and export policies to enable the sending and receiving of

route targets. Configure the BGP peer group within the VRF. Specify AS `20` as the peer AS and specify the IP address of the Fast Ethernet interface on Router CE2 as the neighbor address.

```
[edit routing-instances]
vpn2CE2 {
    instance-type vrf;
    interface fe-0/3/1.0;
    route-distinguisher 1:100;
    vrf-import vpnimport;
    vrf-export vpnexport;
    protocols {
        bgp {
            group To_CE2 {
                peer-as 20;
                neighbor 192.168.6.14;
            }
        }
    }
}
```

3. On Router PE2, configure the RSVP and MPLS protocols to support the LSP. Configure the LSP to ASBR2 and specify the IP address of the logical loopback interface on Router ASBR2. Configure a BGP group. Specify the group type as `internal`. Specify the local address as the logical loopback interface on Router PE2. Specify the neighbor address as the logical loopback interface on the Router ASBR2. Specify the `inet-vpn` address family and `unicast` traffic type to enable BGP to carry IPv4 NLRI for VPN routes. Configure the OSPF protocol. Specify the core-facing SONET interface and specify the logical loopback interface on Router PE2.

```
[edit protocols]
rsvp {
    interface so-0/0/1.0;
    interface lo0.0;
}
mpls {
    label-switched-path To-ASBR2 {
        to 192.0.2.5;
    }
    interface so-0/0/1.0;
    interface lo0.0;
}
bgp {
```

```
    group To_ASBR2 {
        type internal;
        local-address 192.0.2.7;
        neighbor 192.0.2.5 {
            family inet-vpn {
                unicast;
            }
        }
    }
}
ospf {
    traffic-engineering;
    area 0.0.0.0 {
        interface so-0/0/1.0;
        interface lo0.0;
    }
}
```

4. On Router PE2, configure the BGP local autonomous system number.

```
[edit routing-options]
autonomous-system 200;
```

5. On Router PE2, configure a policy to add the VRF route target to the routes being advertised for this VPN.

```
[edit policy-options]
policy-statement vpnexport {
    term 1 {
        from protocol bgp;
        then {
            community add test_comm;
            accept;
        }
    }
    term 2 {
        then reject;
    }
}
```

6. On Router PE2, configure a policy to import routes from BGP that have the `test_comm` community attached.

```
[edit policy-options]
    policy-statement vpnimport {
        term 1 {
            from {
                protocol bgp;
                community test_comm;
            }
            then accept;
        }
        term 2 {
            then reject;
        }
    }
```

7. On Router PE2, define the `test_comm` BGP community with a route target.

```
[edit policy-options]
community test_comm members target:1:100;
```

Configuring Router CE2

Step-by-Step Procedure

1. On Router CE2, configure the IP address and protocol family on the Fast Ethernet interface for the link between Router CE2 and Router PE2. Specify the `inet` address family type.

```
[edit interfaces]
fe-3/0/0 {
    unit 0 {
        family inet {
            address 192.168.6.14/24;
        }
    }
}
```

2. On Router CE2, configure the IP address and protocol family on the loopback interface. Specify the `inet` address family type.

```
[edit interfaces lo0]
lo0 {
    unit 0 {
        family inet {
            address 192.0.2.8/32;
        }
    }
}
```

3. On Router CE2, define a policy named `myroutes` that accepts direct routes.

```
[edit policy-options]
policy-statement myroutes {
    from protocol direct;
    then accept;
}
```

4. On Router CE2, configure a routing protocol. The routing protocol can be a static route, RIP, OSPF, ISIS, or EBGP. In this example, we configure EBGP. Specify AS `200` as the peer AS and specify the BGP neighbor IP address as the Fast Ethernet interface of Router PE2.

```
[edit protocols]
bgp {
    group To_PE2 {
        neighbor 192.168.6.13 {
            export myroutes;
            peer-as 200;
        }
    }
}
```

5. On Router CE2, configure the BGP local autonomous system number.

```
[edit routing-options]
autonomous-system 20;
```

Verifying the VPN Operation

Step-by-Step Procedure

1. Commit the configuration on each router.

   NOTE: The MPLS labels shown in this example will be different than the labels used in your configuration.

2. On Router PE1, display the routes for the `vpn2CE1` routing instance using the `show ospf route` command. Verify that the `192.0.2.1` route is learned from OSPF.

```
user@PE1> show ospf route instance vpn2CE1
Topology default Route Table:

Prefix            Path   Route      NH   Metric  NextHop       Nexthop
                  Type   Type       Type         Interface     addr/label
192.0.2.1           Intra  Router     IP        1  fe-1/2/3.0    198.51.100.1
192.0.2.1/32        Intra  Network    IP        1  fe-1/2/3.0    198.51.100.1
198.51.100.0/24     Intra  Network    IP        1  fe-1/2/3.0    198.51.100.1
```

3. On Router PE1, use the `show route advertising-protocol` command to verify that Router PE1 advertises the `192.0.2.1` route to Router ASBR1 using MP-BGP with the VPN MPLS label.

```
user@PE1> show route advertising-protocol bgp 192.0.2.4 extensive
vpn2CE1.inet.0: 6 destinations, 6 routes (6 active, 0 holddown, 0 hidden)
* 192.0.2.1/32 (1 entry, 1 announced)
 BGP group To_PE1 type Internal
     Route Distinguisher: 1:100
     VPN Label: 299856
     Nexthop: Self
     Flags: Nexthop Change
     MED: 1
     Localpref: 100
     AS path: [100] I
     Communities: target:1:100 rte-type:0.0.0.2:1:0
```

4. On Router ASBR1, use the `show route receive-protocol` command to verify that the router receives and accepts the `192.0.2.1` route and places it in the `To_ASBR2.inet.0` routing table.

```
user@ASBR1> show route receive-protocol bgp 192.0.2.2 extensive
inet.0: 7 destinations, 7 routes (7 active, 0 holddown, 0 hidden)

inet.3: 1 destinations, 1 routes (1 active, 0 holddown, 0 hidden)

To_ASBR2.inet.0: 4 destinations, 4 routes (4 active, 0 holddown, 0 hidden)
* 192.0.2.1/32 (1 entry, 1 announced)
     Route Distinguisher: 1:100
     VPN Label: 299856
     Nexthop: 192.0.2.2
     MED: 1
     Localpref: 100
     AS path: I
     Communities: target:1:100 rte-type:0.0.0.2:1:0

MPLS.0: 4 destinations, 4 routes (4 active, 0 holddown, 0 hidden)

BGP.13VPN.0: 1 destinations, 1 routes (1 active, 0 holddown, 0 hidden)

* 1:100:192.0.2.1/32 (1 entry, 0 announced)
     Route Distinguisher: 1:100
     VPN Label: 299856
     Nexthop: 192.0.2.2
     MED: 1
     Localpref: 100
     AS path: I
     Communities: target:1:100 rte-type:0.0.0.2:1:0
```

5. On Router ASBR1, use the `show route advertising-protocol` command to verify that Router ASBR1 advertises the `192.0.2.1` route to Router ASBR2.

```
user@ASBR1> show route advertising-protocol bgp 192.168.3.8 extensive
To_ASBR2.inet.0: 4 destinations, 4 routes (4 active, 0 holddown, 0 hidden)
* 192.0.2.1/32 (1 entry, 1 announced)
 BGP group To_ASBR2.inet.0 type External
     Nexthop: Self
```

```
     AS path: [100] I
     Communities: target:1:100 rte-type:0.0.0.2:1:0
```

6. On Router ASBR2, use the `show route receive-protocol` command to verify that the router receives and accepts the `192.0.2.1` route and places it in the `To_ASBR1.inet.0` routing table.

```
user@ASBR2> show route receive-protocol bgp 192.168.3.7 extensive
inet.0: 7 destinations, 7 routes (7 active, 0 holddown, 0 hidden)

inet.3: 1 destinations, 1 routes (1 active, 0 holddown, 0 hidden)

To_ASBR1.inet.0: 4 destinations, 4 routes (4 active, 0 holddown, 0 hidden)
* 192.0.2.1/32 (1 entry, 1 announced)
     Accepted
     Nexthop: 192.168.3.7
     AS path: 100 I
     Communities: target:1:100 rte-type:0.0.0.2:1:0

MPLS.0: 4 destinations, 4 routes (4 active, 0 holddown, 0 hidden)

BGP.l3VPN.0: 1 destinations, 1 routes (1 active, 0 holddown, 0 hidden)
```

7. On Router ASBR2, use the `show route advertising-protocol` command to verify that Router ASBR2 advertises the `192.0.2.1` route to Router PE2.

```
user@ASBR2> show route advertising-protocol bgp 192.0.2.7 extensive
To_ASBR1.inet.0: 4 destinations, 4 routes (4 active, 0 holddown, 0 hidden)
* 192.0.2.1/32 (1 entry, 1 announced)
 BGP group To-PE2 type Internal
     Route Distinguisher: 1:100
     VPN Label: 299936
     Nexthop: Self
     Flags: Nexthop Change
     Localpref: 100
     AS path: [200] 100 I
     Communities: target:1:100 rte-type:0.0.0.2:1:0
```

8. On Router PE2, use the `show route receive-protocol` command to verify that the router receives and accepts the `192.0.2.1` route and places it in the `vpn2CE2.inet.0` routing table.

```
user@PE2> show route receive-protocol bgp 192.0.2.5 extensive
inet.0: 12 destinations, 13 routes (12 active, 0 holddown, 0 hidden)

inet.3: 1 destinations, 1 routes (1 active, 0 holddown, 0 hidden)

__juniper_private1__.inet.0: 14 destinations, 14 routes (8 active, 0 holddown, 6 hidden)

__juniper_private2__.inet.0: 1 destinations, 1 routes (0 active, 0 holddown, 1 hidden)

vpn2CE2.inet.0: 5 destinations, 6 routes (5 active, 0 holddown, 0 hidden)
* 192.0.2.1/32 (1 entry, 1 announced)
     Accepted
     Route Distinguisher: 1:100
     VPN Label: 299936
     Nexthop: 192.0.2.5
     Localpref: 100
     AS path: 100 I
     AS path: Recorded
     Communities: target:1:100 rte-type:0.0.0.2:1:0
```

9. On Router PE2, use the `show route advertising-protocol` command to verify that Router PE2 advertises the `192.0.2.1` route to Router CE2 through the `To_CE2` peer group.

```
user@PE2> show route advertising-protocol bgp 192.168.6.14 extensive
vpn2CE2.0: 5 destinations, 6 routes (5 active, 0 holddown, 0 hidden)
* 192.0.2.1/32  (1 entry, 1 announced)
 BGP group To_CE2 type External
     Nexthop: Self
     AS path: [200] 100 I
     Communities: target:1:100 rte-type:0.0.0.2:1:0
```

10. On Router CE2, use the `show route` command to verify that Router CE2 receives the `192.0.2.1` route from Router PE2.

```
user@CE2> show route 192.0.2.1
inet.0: 6 destinations, 6 routes (6 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both
```

```
192.0.2.1/32        *[BGP/170] 00:25:36, localpref 100
                      AS path: 200 100 I
                    > to 192.168.6.13  via fe-3/0/0.0
```

11. On Router CE2, use the `ping` command and specify `192.0.2.8` as the source of the ping packets to verify connectivity with Router CE1.

```
user@CE2> ping 192.0.2.1 source 192.0.2.8
PING 192.0.2.1 (192.0.2.1): 56 data bytes
64 bytes from 192.0.2.1: icmp_seq=0 ttl=58 time=4.672 ms
64 bytes from 192.0.2.1: icmp_seq=1 ttl=58 time=10.480 ms
64 bytes from 192.0.2.1: icmp_seq=2 ttl=58 time=10.560 ms
```

12. On Router PE2, use the `show route` command to verify that the traffic is sent with an inner label of `299936` and a top label of `299776`.

```
user@PE2> show route 192.0.2.1 detail

vpn2CE2.inet.0: 5 destinations, 6 routes (5 active, 0 holddown, 0 hidden)
192.0.2.1/32 (1 entry, 1 announced)
        *BGP    Preference: 170/-101
                Route Distinguisher: 1:100
                Next hop type: Indirect
                Next-hop reference count: 6
                Source: 192.0.2.5
                Next hop type: Router, Next hop index: 648
                Next hop: via so-0/0/1.0 weight 0x1, selected
                Label-switched-path To-ASBR2
                Label operation: Push 299936, Push 299776(top)
                Protocol next hop: 192.0.2.5
                Push 299984
                Indirect next hop: 8c6109c 262143
                State: <Secondary Active Int Ext>
                Local AS:   200 Peer AS:   200
                Age: 3:37       Metric2: 2
                Task: BGP_200.192.0.2.5+179
                Announcement bits (3): 0-RT 1-KRT 2-BGP RT Background
                AS path: 100 I
                AS path: Recorded
                Communities: target:1:100 rte-type:0.0.0.2:1:0
```

```
                Accepted
                VPN Label: 299984
                Localpref: 100
                Router ID: 192.0.2.5
                Primary Routing Table BGP.l3VPN.0
```

13. On Router ASBR2, use the `show route table` command to verify that Router ASBR2 receives the traffic.

```

user@ASBR2# show route table mpls.0 detail
299936 (1 entry, 1 announced)
        *VPN    Preference: 170
                Next hop type: Router, Next hop index: 649
                Next-hop reference count: 2
                Source: 192.168.3.7                Next hop: 192.168.3.7 via ge-0/1/1.0,
selected
                Label operation: Pop
                State: <Active Int Ext>
                Local AS:   200
                Age: 9:54
                Task: BGP RT Background
                Announcement bits (1): 0-KRT
                AS path: 100 I
                Ref Cnt: 1
                Communities: target:1:100 rte-type:0.0.0.2:1:0

```

14. On Router ASBR2, use the `show route table` command to verify that Router ASBR2 receives the traffic.

```

user@ASBR2# show route 192.0.2.1  detail
To_ASBR1.inet.0: 4 destinations, 4 routes (4 active, 0 holddown, 0 hidden)
192.0.2.1/32  (1 entry, 1 announced)
        *BGP    Preference: 170/-101
                Next hop type: Router, Next hop index: 576
                Next-hop reference count: 3
                Source: 192.168.3.7
                Next hop: 192.168.3.7  via ge-0/1/1.0, selected
                State: <Active Ext>
                Peer AS:   100
                Age: 13:07
```

```
                Task: BGP_192.168.3.7+53372
                Announcement bits (2): 0-KRT 1-BGP RT Background
                AS path: 100 I
                Communities: target:1:100 rte-type:0.0.0.2:1:0
                Accepted
                Localpref: 100
                Router ID: 192.168.3.7
```

15. On Router ASBR1, use the `show route` command to verify that ASBR1 sends traffic toward PE1 with the top label `299792` and VPN label `299856`.

```
user@ASBR1# show route 192.0.2.1 detail
To_ASBR2.inet.0: 4 destinations, 4 routes (4 active, 0 holddown, 0 hidden)
192.0.2.1/24 (1 entry, 1 announced)
        *BGP    Preference: 170/-101
                Route Distinguisher: 1:100
                Next hop type: Indirect
                Next-hop reference count: 3
                Source: 192.0.2.2
                Next hop type: Router, Next hop index: 669
                Next hop: 192.168.2.5 via ge-0/0/0.0 weight 0x1, selected
                Label-switched-path To_PE1
                Label operation: Push 299856, Push 299792(top)
                Protocol next hop: 192.0.2.2                Push 299856
                Indirect next hop: 8af70a0 262143
                State: <Secondary Active Int Ext>
                Local AS:   100 Peer AS:   100
                Age: 12:15      Metric: 1       Metric2: 2
                Task: BGP_100.192.0.2.2+58065
                Announcement bits (2): 0-KRT 1-BGP RT Background
                AS path: I
                Communities: target:1:100 rte-type:0.0.0.2:1:0
                VPN Label: 299856
                Localpref: 100
                Router ID: 192.0.2.2
                Primary Routing Table BGP.l3VPN.0
```

16. On Router PE1, use the `show route table` command to verify that Router PE1 receives the traffic with label `299856`, pops the label,l and the traffic is sent toward Router CE1 through interface `fe-1/2/3.0`.

```
lab@PE1# show route table mpls.0 detail
299856 (1 entry, 1 announced)
        *VPN    Preference: 170
                Next hop type: Router, Next hop index: 666
                Next-hop reference count: 2
                Next hop: 198.51.100.8 via fe-1/2/3.0, selected
                Label operation: Pop
                State: <Active Int Ext>
                Local AS:   100
                Age: 17:38
                Task: BGP RT Background
                Announcement bits (1): 0-KRT
                AS path: I
                Ref Cnt: 1
                Communities: rte-type:0.0.0.2:1:0
```

17. On Router PE1, use the `show route` command to verify that PE1 receives the traffic after the top label is popped by Router P and the traffic is sent toward Router CE1 through interface `fe-1/2/3.0`.

```
lab@PE1# show route 192.0.2.1 detail

 vpn2CE1.inet.0: 6 destinations, 6 routes (6 active, 0 holddown, 0 hidden)
192.0.2.1/32 (1 entry, 1 announced)
        *OSPF   Preference: 10
                Next hop type: Router, Next hop index: 634
                Next-hop reference count: 3
                Next hop: 198.51.100.8 via fe-1/2/3.0, selected
                State: <Active Int>
                Age: 18:42      Metric: 1
                Area: 0.0.0.2
                Task: VPN2alice-OSPFv2
                Announcement bits (2): 2-KRT 3-BGP RT Background
                AS path: I
                Communities: rte-type:0.0.0.2:1:0
```

Example: Configuring Interprovider Layer 3 VPN Option B

IN THIS SECTION

Interprovider Layer 3 VPN Option B provides interprovider EBGP redistribution of labeled VPN-IPv4 routes from AS to neighboring AS. This solution is considered to be more scalable than Option A, but not as scalable as Option C.

This example provides a step-by-step procedure to configure interprovider layer 3 VPN option B, which is one of the recommended implementations of an MPLS VPN for a customer that has more than one AS, but not all of the customer's ASs can be serviced by the same service provider. It is organized in the following sections:

Requirements

This example uses the following hardware and software components:

- Junos OS Release 9.5 or later.
    - This example has been recently updated and revalidated on Junos OS Release 21.1R1.
- Eight M Series, T Series, TX Series, QFX10000, or MX Series Juniper Networks routers.

Configuration Overview and Topology

IN THIS SECTION

Interprovider layer 3 VPN option B is a somewhat scalable solution to the problem of providing VPN services to a customer that has different sites, not all of which can use the same service provider. *RFC 4364*, section 10, refers to this method as interprovider EBGP redistribution of labeled VPN-IPv4 routes from AS to neighboring AS.

In the topology shown in Figure 1, the following events occur:

- The PE routers use IBGP to redistribute labeled VPN-IPv4 routes to an ASBR.
- The ASBR then uses EBGP to redistribute those labeled VPN-IPv4 routes to an ASBR in another AS, which distributes them to the PE routers in that AS.
- Labeled VPN-IPv4 routes are distributed between ASBR routers on each site. There is no need to define a separate VPN routing and forwarding instance (VRF) for each common VPN that resides on two different SPs.
- Router PE2 distributes VPN-IPv4 routes to Router ASBR2 using MP-IBGP.
- Router ASBR2 distributes these labeled VPN-IPv4 routes to Router ASBR1, using the MP-EBGP session between them.
- Router ASBR1 redistributes those routes to Router PE1, using MP-IBGP. Each time a label is advertised, routers change the next-hop information and labels.
- An MPLS path is established between Router PE1 and Router PE2. This path enables changing of the next-hop attribute for the routes that are learned from the neighbor SP router and map the incoming label for the given routes to the outgoing label advertised to PE routers in the internal network.
- The ingress PE router inserts two labels onto the IP packet coming from the end customer. The inner label is for the VPN-IPv4 routes learned from internal ASBRs and the outer label is for the route to the internal ASBR, obtained through resource reservation protocol (RSVP) or label distribution protocol (LDP).
- When a packet arrives at the ASBR, it removes the outer label (when explicit-null signaling is used; otherwise, penultimate hop-popping (PHP) pops the label) and swaps the inner label with the label obtained from the neighbor ASBR through MP-EBGP label and prefix advertisements.
- The second ASBR swaps the VPN-IPv4 label and pushes another label to reach the PE router in its own AS.
- The remaining process is the same as for a regular VPN.

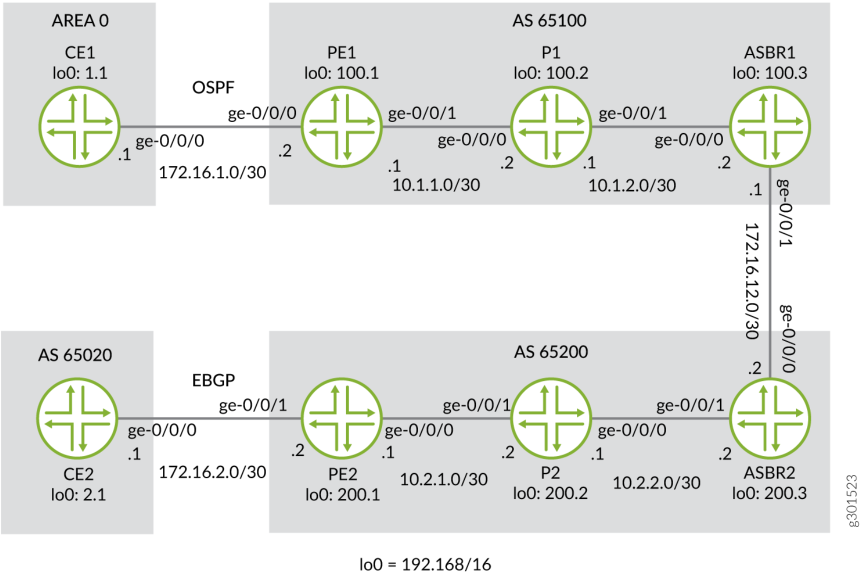
NOTE: In this solution, ASBR routers keep all VPN-IPv4 routes in the routing information base (RIB), and the labels associated with the prefixes are kept in the forwarding information base (FIB). Because the RIB and FIB tables can take occupy much of the respective allocated memory, this solution is not very scalable for an interprovider VPN.

If a transit SP is used between SP1 and SP2, the transit SP also has to keep all VPN-IPv4 routes in the RIB and the corresponding labels in the FIB. The ASBRs at the transit SP have the same functionality as ASBRs in the SP1 or SP2 networks in this solution.

Topology

The topology of the network is shown in Figure 48 on page 481.

Figure 48: Physical Topology of Interprovider Layer 3 VPN Option B



Configuration

IN THIS SECTION

NOTE: The procedure presented here is written with the assumption that the reader is already familiar with MPLS MVPN configuration. This example focuses on explaining the unique configuration required for carrier-of-carriers solutions for VPN services to different sites.

To configure layer 3 VPN option B, perform the following tasks:

Configuring Router CE1

Step-by-Step Procedure

1. On Router CE1, configure the IP address and protocol family on the logical loopback interface and the Gigabit Ethernet interface for the link between Router CE1 and Router PE1. Specify the `inet` address family type.

```
user@CE1#
set interfaces ge-0/0/0 description to_PE1
set interfaces ge-0/0/0 unit 0 family inet address 172.16.1.1/30
set interfaces lo0 unit 0 family inet address 192.168.1.1/32
```

2. On Router CE1, configure the router ID.

```
user@CE1#
set routing-options router-id 192.168.1.1
```

3. On Router CE1, configure a routing protocol. Include the logical interface for the link between Router CE1 and Router PE1 and the logical loopback interface of Router CE1. The routing protocol can be a static route, RIP, OSPF, ISIS, or EBGP. In this example we configure OSPF.

```
user@CE1#
set protocols ospf area 0.0.0.0 interface ge-0/0/0.0
set protocols ospf area 0.0.0.0 interface lo0.0
```

Configuring Router PE1

Step-by-Step Procedure

1. On Router PE1, configure IPv4 addresses on the Gigabit Ethernet and logical loopback interfaces. Specify the `inet` address family on all of the interfaces. Specify the `mpls` address family on the core-facing interface.

```
user@PE1#
set interfaces ge-0/0/0 description to_CE1
set interfaces ge-0/0/0 unit 0 family inet address 172.16.1.2/30
set interfaces ge-0/0/1 description to_P1
set interfaces ge-0/0/1 unit 0 family inet address 10.1.1.1/30
set interfaces ge-0/0/1 unit 0 family mpls
set interfaces lo0 unit 0 family inet address 192.168.100.1/32
```

2. On Router PE1, configure a VRF routing instance. Specify the `vrf` instance type and specify the customer-facing interface. Configure a route distinguisher to create a unique VPN-IPv4 address prefix. Apply the VRF import and export policies to enable the sending and receiving of route targets. Configure the OSPF protocol within the VRF. Specify the customer-facing interface and specify the export policy to export BGP routes into OSPF.

```
user@PE1#
set routing-instances to_CE1 instance-type vrf
set routing-instances to_CE1 protocols ospf area 0.0.0.0 interface ge-0/0/0.0
set routing-instances to_CE1 protocols ospf export bgp-to-ospf
set routing-instances to_CE1 interface ge-0/0/0.0
set routing-instances to_CE1 route-distinguisher 192.168.100.1:1
set routing-instances to_CE1 vrf-import vpnimport
set routing-instances to_CE1 vrf-export vpnexport
```

3. On Router PE1, configure the RSVP and MPLS protocols to support the label-switched path (LSP). Configure the LSP to Router ASBR1 and specify the IP address of the logical loopback interface on Router ASBR1. Configure a BGP group. Specify the group type as `internal`. Specify the local address as the logical loopback interface on Router PE1. Specify the neighbor address as the logical loopback interface on Router ASBR1. Specify the `inet-vpn` address family and `unicast` traffic type to enable BGP

to carry IPv4 network layer reachability information (NLRI) for VPN routes. Configure the OSPF protocol. Specify the core-facing interface and specify the logical loopback interface on Router PE1.

```
user@PE1#
set protocols bgp group to-ASBR1 type internal
set protocols bgp group to-ASBR1 local-address 192.168.100.1
set protocols bgp group to-ASBR1 neighbor 192.168.100.3 family inet-vpn unicast
set protocols mpls label-switched-path to-ASBR1 to 192.168.100.3
set protocols mpls interface ge-0/0/1.0
set protocols mpls interface lo0.0
set protocols ospf traffic-engineering
set protocols ospf area 0.0.0.0 interface ge-0/0/1.0
set protocols ospf area 0.0.0.0 interface lo0.0
set protocols rsvp interface ge-0/0/1.0
set protocols rsvp interface lo0.0
```

4. On Router PE1, configure the BGP local autonomous system number and router ID.

```
user@PE1#
set routing-options router-id 192.168.100.1
set routing-options autonomous-system 65100
```

5. On Router PE1, configure a policy to export the BGP routes into OSPF.

```
user@PE1#
set policy-options policy-statement bgp-to-ospf term 1 from protocol bgp
set policy-options policy-statement bgp-to-ospf term 1 then accept
set policy-options policy-statement bgp-to-ospf term 2 then reject
```

6. On Router PE1, configure a policy to add the VRF route target to the routes being advertised from CE1.

```
user@PE1#
set policy-options policy-statement vpnexport term 1 from protocol ospf
set policy-options policy-statement vpnexport term 1 then community add pe1_comm
set policy-options policy-statement vpnexport term 1 then accept
set policy-options policy-statement vpnexport term 2 then reject
```

7. On Router PE1, configure a policy to import routes from PE2 that have the `pe2_comm` community attached.

   ```
   user@PE1#
   set policy-options policy-statement vpnimport term 1 from protocol bgp
   set policy-options policy-statement vpnimport term 1 from community pe2_comm
   set policy-options policy-statement vpnimport term 1 then accept
   set policy-options policy-statement vpnimport term 2 then reject
   ```

8. On Router PE1, define the `pe1_comm` BGP community with a route target to apply to the *vpnexport* policy and define the `pe2_comm` BGP community with a route target to apply to the *vpnimport* policy.

   ```
   user@PE1#
   set policy-options community pe1_comm members target:65100:1
   set policy-options community pe2_comm members target:65200:1
   ```

Configuring Router P1

Step-by-Step Procedure

1. On Router P1, configure IP addresses for the Gigabit Ethernet interfaces. Enable the interfaces to process the `inet` and `mpls` address families. Configure the IP addresses for the `lo0.0` loopback interface and enable the interface to process the `inet` address family.

   ```
   user@P1#
   set interfaces ge-0/0/0 description to_PE1
   set interfaces ge-0/0/0 unit 0 family inet address 10.1.1.2/30
   set interfaces ge-0/0/0 unit 0 family mpls
   set interfaces ge-0/0/1 description to_ASBR1
   set interfaces ge-0/0/1 unit 0 family inet address 10.1.2.1/30
   set interfaces ge-0/0/1 unit 0 family mpls
   set interfaces lo0 unit 0 family inet address 192.168.100.2/32
   ```

2. On Router P1, configure the RSVP and MPLS protocols to support the LSP. Specify the Gigabit Ethernet interfaces.

Configure the OSPF protocol. Specify the Gigabit Ethernet interfaces and specify the logical loopback interface. Enable OSPF to support traffic engineering extensions.

```
user@P1#
set protocols mpls interface ge-0/0/0.0
set protocols mpls interface ge-0/0/1.0
set protocols mpls interface lo0.0
set protocols ospf traffic-engineering
set protocols ospf area 0.0.0.0 interface ge-0/0/0.0
set protocols ospf area 0.0.0.0 interface ge-0/0/1.0
set protocols ospf area 0.0.0.0 interface lo0.0
set protocols rsvp interface ge-0/0/0.0
set protocols rsvp interface ge-0/0/1.0
set protocols rsvp interface lo0.0
```

Configuring Router ASBR1

Step-by-Step Procedure

1. On Router ASBR1, configure IP addresses for the Gigabit Ethernet interfaces. Enable the interfaces to process the `inet` and `mpls` addresses families. Configure the IP addresses for the `lo0.0` loopback interface and enable the interface to process the `inet` address family.

```
user@ASBR1#
set interfaces ge-0/0/0 description to_P1
set interfaces ge-0/0/0 unit 0 family inet address 10.1.2.2/30
set interfaces ge-0/0/0 unit 0 family mpls
set interfaces ge-0/0/1 description to_ASBR2
set interfaces ge-0/0/1 unit 0 family inet address 172.16.12.1/30
set interfaces ge-0/0/1 unit 0 family mpls
set interfaces lo0 unit 0 family inet address 192.168.100.3/32
```

2. On Router ASBR1, configure the RSVP and MPLS protocols to support the LSP by specifying the Gigabit Ethernet interface facing the P1 router and the `lo0.0` logical loopback interface.

   Configure the OSPF protocol by specifying the Gigabit Ethernet interface that is facing the P1 router and the logical loopback interface. Enable OSPF to support traffic engineering extensions.

```
user@ASBR1#
set protocols mpls label-switched-path to-PE1 to 192.168.100.1
```

```
set protocols mpls interface ge-0/0/0.0
set protocols mpls interface lo0.0
set protocols ospf traffic-engineering
set protocols ospf area 0.0.0.0 interface ge-0/0/0.0
set protocols ospf area 0.0.0.0 interface lo0.0
set protocols rsvp interface ge-0/0/0.0
set protocols rsvp interface lo0.0
```

3. On Router ASBR1, create the `to-PE1` internal BGP peer group. Specify the local IP peer address as the local `lo0.0` address. Specify the neighbor IP peer address as the `lo0.0` interface address of Router PE1.

```
user@ASBR1#
set protocols bgp group to-PE1 type internal
set protocols bgp group to-PE1 local-address 192.168.100.3
set protocols bgp group to-PE1 neighbor 192.168.100.1 family inet-vpn unicast
```

4. On Router ASBR1, create the `to-ASBR2` external BGP peer group. Enable the router to use BGP to advertise NLRI for unicast routes. Specify the neighbor IP peer address as the Gigabit Ethernet interface address of Router ASBR2.

```
user@ASBR1#
set protocols bgp group to-ASBR2 type external
set protocols bgp group to-ASBR2 family inet-vpn unicast
set protocols bgp group to-ASBR2 neighbor 172.16.12.2 peer-as 65200
```

5. On Router ASBR1, configure the BGP local autonomous system number the router ID.

```
user@ASBR1#
set routing-options router-id 192.168.100.3
set routing-options autonomous-system 65100
```

Configuring Router ASBR2

Step-by-Step Procedure

1. On Router ASBR2, configure IP addresses for the Gigabit Ethernet interfaces. Enable the interfaces to process the `inet` and `mpls` address families. Configure the IP address for the `lo0.0` loopback interface and enable the interface to process the `inet` address family.

```
user@ASBR2#
set interfaces ge-0/0/0 description to_ASBR1
set interfaces ge-0/0/0 unit 0 family inet address 172.16.12.2/30
set interfaces ge-0/0/0 unit 0 family mpls
set interfaces ge-0/0/1 description to_P2
set interfaces ge-0/0/1 unit 0 family inet address 10.2.2.2/30
set interfaces ge-0/0/1 unit 0 family mpls
set interfaces lo0 unit 0 family inet address 192.168.200.3/32
```

2. On Router ASBR2, configure the RSVP and MPLS protocols to support the LSP by specifying the Gigabit Ethernet interface that is facing the P2 router.

   Configure the OSPF protocol by specifying the Gigabit Ethernet interface that is facing the P2 router and the logical loopback interface. Enable OSPF to support traffic engineering extensions.

```
user@ASBR2#
set protocols mpls label-switched-path to-PE2 to 192.168.200.1
set protocols mpls interface ge-0/0/1.0
set protocols mpls interface lo0.0
set protocols ospf traffic-engineering
set protocols ospf area 0.0.0.0 interface ge-0/0/1.0
set protocols ospf area 0.0.0.0 interface lo0.0
set protocols rsvp interface ge-0/0/1.0
set protocols rsvp interface lo0.0
```

3. On Router ASBR2, create the `to-PE2` internal BGP peer group. Specify the local IP peer address as the local `lo0.0` address. Specify the neighbor IP peer address as the `lo0.0` interface address of Router PE2.

```
user@ASBR2#
set protocols bgp group to-PE2 type internal
set protocols bgp group to-PE2 local-address 192.168.200.3
set protocols bgp group to-PE2 neighbor 192.168.200.1 family inet-vpn unicast
```

4. On Router ASBR2, create the `to-ASBR1` external BGP peer group. Enable the router to use BGP to advertise NLRI for unicast routes. Specify the neighbor IP peer address as the Gigabit Ethernet interface on Router ASBR1.

```
user@ASBR2#
set protocols bgp group to-ASBR1 type external
set protocols bgp group to-ASBR1 family inet-vpn unicast
set protocols bgp group to-ASBR1 neighbor 172.16.12.1 peer-as 65100
```

5. On Router ASBR2, configure the BGP local autonomous system number and the router ID.

```
user@ASBR2#
set routing-options router-id 192.168.200.3
set routing-options autonomous-system 65200
```

Configuring Router P2

Step-by-Step Procedure

1. On Router P2, configure IP addresses for the Gigabit Ethernet interfaces. Enable the interfaces to process the `inet` and `mpls` address families. Configure the IP address for the `lo0.0` loopback interface and enable the interface to process the `inet` address family.

```
user@P2#
set interfaces ge-0/0/0 description to_ASBR2
set interfaces ge-0/0/0 unit 0 family inet address 10.2.2.1/30
set interfaces ge-0/0/0 unit 0 family mpls
set interfaces ge-0/0/1 description to_PE2
set interfaces ge-0/0/1 unit 0 family inet address 10.2.1.2/30
set interfaces ge-0/0/1 unit 0 family mpls
set interfaces lo0 unit 0 family inet address 192.168.200.2/32
```

2. On Router P2, configure the RSVP and MPLS protocols to support the LSP. Specify the Gigabit Ethernet interfaces.

Configure the OSPF protocol. Specify the Gigabit Ethernet interfaces and specify the logical loopback interface. Enable OSPF to support traffic engineering extensions.

```
user@P2#
set protocols mpls interface ge-0/0/0.0
set protocols mpls interface ge-0/0/1.0
set protocols mpls interface lo0.0
set protocols ospf traffic-engineering
set protocols ospf area 0.0.0.0 interface ge-0/0/0.0
set protocols ospf area 0.0.0.0 interface ge-0/0/1.0
set protocols ospf area 0.0.0.0 interface lo0.0
set protocols rsvp interface ge-0/0/0.0
set protocols rsvp interface ge-0/0/1.0
set protocols rsvp interface lo0.0
```

Configuring Router PE2

Step-by-Step Procedure

1. On Router PE2, configure IPv4 addresses on the Gigabit Ethernet and logical loopback interfaces. Specify the `inet` address family on all of the interfaces. Specify the `mpls` address family on the Gigabit Ethernet interfaces.

```
user@PE2#
set interfaces ge-0/0/0 description to_P2
set interfaces ge-0/0/0 unit 0 family inet address 10.2.1.1/30
set interfaces ge-0/0/0 unit 0 family mpls
set interfaces ge-0/0/1 description to_CE2
set interfaces ge-0/0/1 unit 0 family inet address 172.16.2.2/30
set interfaces lo0 unit 0 family inet address 192.168.200.1/32
```

2. On Router PE2, configure a VRF routing instance. Specify the `vrf` instance type and specify the customer-facing interface. Configure a route distinguisher to create a unique VPN-IPv4 address prefix. Apply the VRF import and export policies to enable the sending and receiving of route targets. Configure the BGP peer group within the VRF. Specify AS `65020` as the peer AS and specify the IP address of the Gigabit Ethernet interface on Router CE1 as the neighbor address.

```
user@PE2#
set routing-instances to_CE2 instance-type vrf
```

```
set routing-instances to_CE2 protocols bgp group to_CE2 peer-as 65020
set routing-instances to_CE2 protocols bgp group to_CE2 neighbor 172.16.2.1
set routing-instances to_CE2 interface ge-0/0/1.0
set routing-instances to_CE2 route-distinguisher 192.168.200.1:1
set routing-instances to_CE2 vrf-import vpnimport
set routing-instances to_CE2 vrf-export vpnexport
```

3. On Router PE2, configure the RSVP and MPLS protocols to support the LSP. Configure the LSP to ASBR2 and specify the IP address of the logical loopback interface on Router ASBR2. Configure a BGP group. Specify the group type as `internal`. Specify the local address as the logical loopback interface on Router PE2. Specify the neighbor address as the logical loopback interface on the Router ASBR2. Specify the `inet-vpn` address family and `unicast` traffic type to enable BGP to carry IPv4 NLRI for VPN routes. Configure the OSPF protocol. Specify the core-facing interface and the logical loopback interface on Router PE2.

```
user@PE2#
set protocols bgp group to-ASBR2 type internal
set protocols bgp group to-ASBR2 local-address 192.168.200.1
set protocols bgp group to-ASBR2 neighbor 192.168.200.3 family inet-vpn unicast
set protocols mpls label-switched-path to-ASBR2 to 192.168.200.3
set protocols mpls interface ge-0/0/0.0
set protocols mpls interface lo0.0
set protocols ospf traffic-engineering
set protocols ospf area 0.0.0.0 interface ge-0/0/0.0
set protocols ospf area 0.0.0.0 interface lo0.0
set protocols rsvp interface ge-0/0/0.0
set protocols rsvp interface lo0.0
```

4. On Router PE2, configure the BGP local autonomous system number and the router ID.

```
user@PE2#
set routing-options router-id 192.168.200.1
set routing-options autonomous-system 65200
```

5. On Router PE2, configure a policy to add the VRF route target to the routes being advertised from CE2.

```
user@PE2#
set policy-options policy-statement vpnexport term 1 from protocol bgp
set policy-options policy-statement vpnexport term 1 then community add pe2_comm
```

```
set policy-options policy-statement vpnexport term 1 then accept
set policy-options policy-statement vpnexport term 2 then reject
```

6. On Router PE2, configure a policy to import routes from PE1 that have the `pe1_comm` community attached.

```
user@PE2#
set policy-options policy-statement vpnimport term 1 from protocol bgp
set policy-options policy-statement vpnimport term 1 from community pe1_comm
set policy-options policy-statement vpnimport term 1 then accept
set policy-options policy-statement vpnimport term 2 then reject
```

7. On Router PE2, define the `pe2_comm` BGP community with a route target to apply to the *vpnexport* policy and define the `pe1_comm` BGP community with a route target to apply to the *vpnimport* policy

```
user@PE2#
set policy-options community pe1_comm members target:65100:1
set policy-options community pe2_comm members target:65200:1
```

Configuring Router CE2

Step-by-Step Procedure

1. On Router CE2, configure the IP address and protocol family on the logical loopback interface and the Gigabit Ethernet interface for the link between Router CE2 and Router PE2. Specify the `inet` address family type.

```
user@CE2#
set interfaces ge-0/0/0 description to_PE2
set interfaces ge-0/0/0 unit 0 family inet address 172.16.2.1/30
set interfaces lo0 unit 0 family inet address 192.168.2.1/32
```

2. On Router CE2, define a policy named `loopback` that matches on the loopback address for CE2.

```
user@CE2#
set policy-options policy-statement loopback term 1 from route-filter 192.168.2.1/32 exact
set policy-options policy-statement loopback term 1 then accept
```

3. On Router CE2, configure a routing protocol. The routing protocol can be a static route, RIP, OSPF, ISIS, or EBGP. In this example, we configure EBGP. Specify AS `65200` as the peer AS and specify the BGP neighbor IP address as the Gigabit Ethernet interface of Router PE2. Include the `export` statement.

```
user@CE2#
set protocols bgp group to_PE2 export loopback
set protocols bgp group to_PE2 peer-as 65200
set protocols bgp group to_PE2 neighbor 172.16.2.2
```

4. On Router CE2, configure the BGP local autonomous system number and the router ID.

```
user@CE2#
set routing-options router-id 192.168.2.1
set routing-options autonomous-system 65020
```

Verifying the VPN Operation

Step-by-Step Procedure

1. Commit the configuration on each router.

> **NOTE**: The MPLS labels shown in this example will be different than the labels used in your configuration.

2. On Router PE1, display the routes for the `to_CE1` routing instance using the `show ospf route` command. Verify that the `192.168.1.1` route is learned from OSPF.

```
user@PE1> show ospf route instance to_CE1
Topology default Route Table:

Prefix             Path  Route      NH       Metric NextHop       Nexthop
                   Type  Type       Type            Interface     Address/LSP
192.168.1.1        Intra Router     IP            1 ge-0/0/0.0    172.16.1.1
172.16.1.0/30      Intra Network    IP            1 ge-0/0/0.0
192.168.1.1/32     Intra Network    IP            1 ge-0/0/0.0    172.16.1.1
```

3. On Router PE1, use the `show route advertising-protocol` command to verify that Router PE1 advertises the `192.168.1.1` route to Router ASBR1 using MP-BGP with the VPN MPLS label.

```
user@PE1> show route advertising-protocol bgp 192.168.100.3 extensive
to_CE1.inet.0: 5 destinations, 5 routes (5 active, 0 holddown, 0 hidden)
* 192.168.1.1/32 (1 entry, 1 announced)
 BGP group to-ASBR1 type Internal
     Route Distinguisher: 192.168.100.1:1
     VPN Label: 299808
     Nexthop: Self
     Flags: Nexthop Change
     MED: 1
     Localpref: 100
     AS path: [65100] I
     Communities: target:65100:1 rte-type:0.0.0.0:1:0
```

4. On Router ASBR1, use the `show route receive-protocol` command to verify that the router receives and accepts the `192.168.1.1` route and places it in the `bgp.l3vpn.0` routing table.

```
user@ASBR1> show route receive-protocol bgp 192.168.100.1 extensive
inet.0: 15 destinations, 15 routes (15 active, 0 holddown, 0 hidden)

inet.3: 1 destinations, 1 routes (1 active, 0 holddown, 0 hidden)

mpls.0: 6 destinations, 6 routes (6 active, 0 holddown, 0 hidden)

bgp.l3vpn.0: 2 destinations, 2 routes (2 active, 0 holddown, 0 hidden)
* 192.168.100.1:1:192.168.1.1/32 (1 entry, 1 announced)
     Accepted
     Route Distinguisher: 192.168.100.1:1
     VPN Label: 299808
     Nexthop: 192.168.100.1
     MED: 1
     Localpref: 100
     AS path: I
     Communities: target:65100:1 rte-type:0.0.0.0:1:0
```

5. On Router ASBR1, use the `show route advertising-protocol` command to verify that Router ASBR1 advertises the `192.168.1.1` route to Router ASBR2.

```
user@ASBR1> show route advertising-protocol bgp 172.16.12.2 extensive
bgp.l3vpn.0: 2 destinations, 2 routes (2 active, 0 holddown, 0 hidden)
* 192.168.100.1:1:192.168.1.1/32 (1 entry, 1 announced)
 BGP group to-ASBR2 type External
     Route Distinguisher: 192.168.100.1:1
     VPN Label: 299824
     Nexthop: Self
     Flags: Nexthop Change
     AS path: [65100] I
     Communities: target:65100:1 rte-type:0.0.0.0:1:0
```

6. On Router ASBR2, use the `show route receive-protocol` command to verify that the router receives and accepts the `192.168.1.1` route and places it in the `bgp.l3vpn.0` routing table.

```
user@ASBR2> show route receive-protocol bgp 172.16.12.1 extensive
inet.0: 15 destinations, 15 routes (15 active, 0 holddown, 0 hidden)

inet.3: 1 destinations, 1 routes (1 active, 0 holddown, 0 hidden)

mpls.0: 6 destinations, 6 routes (6 active, 0 holddown, 0 hidden)

bgp.l3vpn.0: 2 destinations, 2 routes (2 active, 0 holddown, 0 hidden)
* 192.168.100.1:1:192.168.1.1/32 (1 entry, 1 announced)
     Accepted
     Route Distinguisher: 192.168.100.1:1
     VPN Label: 299824
     Nexthop: 172.16.12.1
     AS path: 65100 I
     Communities: target:65100:1 rte-type:0.0.0.0:1:0
```

7. On Router ASBR2, use the `show route advertising-protocol` command to verify that Router ASBR2 advertises the `192.168.1.1` route to Router PE2.

```
user@ASBR2> show route advertising-protocol bgp 192.168.200.1 extensive
bgp.l3vpn.0: 2 destinations, 2 routes (2 active, 0 holddown, 0 hidden)
* 192.168.100.1:1:192.168.1.1/32 (1 entry, 1 announced)
 BGP group to-PE2 type Internal
```

```
     Route Distinguisher: 192.168.100.1:1
     VPN Label: 299824
     Nexthop: Self
     Flags: Nexthop Change
     Localpref: 100
     AS path: [65200] 65100 I
     Communities: target:65100:1 rte-type:0.0.0.0:1:0
```

8. On Router PE2, use the `show route receive-protocol` command to verify that the router receives and accepts the `192.168.1.1` route and places it in the `to_CE2.inet.0` routing table.

```
user@PE2> show route receive-protocol bgp 192.168.200.3 extensive
inet.0: 13 destinations, 13 routes (13 active, 0 holddown, 0 hidden)

inet.3: 1 destinations, 1 routes (1 active, 0 holddown, 0 hidden)
to_CE2.inet.0: 4 destinations, 4 routes (4 active, 0 holddown, 0 hidden)
* 192.168.1.1/32 (1 entry, 1 announced)
     Import Accepted
     Route Distinguisher: 192.168.100.1:1
     VPN Label: 299824
     Nexthop: 192.168.200.3
     Localpref: 100
     AS path: 65100 I
     Communities: target:65100:1 rte-type:0.0.0.0:1:0

mpls.0: 5 destinations, 5 routes (5 active, 0 holddown, 0 hidden)

bgp.l3vpn.0: 1 destinations, 1 routes (1 active, 0 holddown, 0 hidden)

* 192.168.100.1:1:192.168.1.1/32 (1 entry, 0 announced)
     Import Accepted
     Route Distinguisher: 192.168.100.1:1
     VPN Label: 299824
     Nexthop: 192.168.200.3
     Localpref: 100
     AS path: 65100 I
     Communities: target:65100:1 rte-type:0.0.0.0:1:0

inet6.0: 1 destinations, 1 routes (1 active, 0 holddown, 0 hidden)

to_CE2.inet6.0: 1 destinations, 1 routes (1 active, 0 holddown, 0 hidden)
```

9. On Router PE2, use the `show route advertising-protocol` command to verify that Router PE2 advertises the `192.168.1.1` route to Router CE2 through the `to_CE2` peer group.

```
user@PE2> show route advertising-protocol bgp 172.16.2.1 extensive
to_CE2.inet.0: 4 destinations, 4 routes (4 active, 0 holddown, 0 hidden)
* 192.168.1.1/32 (1 entry, 1 announced)
 BGP group to_CE2 type External
     Nexthop: Self
     AS path: [65200] 65100 I
     Communities: target:65100:1 rte-type:0.0.0.0:1:0
```

10. On Router CE2, use the `show route` command to verify that Router CE2 receives the `192.168.1.1` route from Router PE2.

```
user@CE2> show route 192.168.1.1
inet.0: 10 destinations, 10 routes (10 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

192.168.1.1/32     *[BGP/170] 6d 02:09:53, localpref 100
                      AS path: 65200 65100 I, validation-state: unverified
                    >  to 172.16.2.2 via ge-0/0/0.0
```

11. On Router CE2, use the `ping` command and specify `192.168.2.1` as the source of the ping packets to verify connectivity with Router CE1.

```
user@CE2> ping 192.168.1.1 source 192.168.2.1 count 2
PING 192.168.1.1 (192.168.1.1): 56 data bytes
64 bytes from 192.168.1.1: icmp_seq=0 ttl=58 time=27.008 ms
64 bytes from 192.168.1.1: icmp_seq=1 ttl=58 time=40.004 ms

--- 192.168.1.1 ping statistics ---
2 packets transmitted, 2 packets received, 0% packet loss
round-trip min/avg/max/stddev = 27.008/33.506/40.004/6.498 ms
```

NOTE: To ping end-to-end without sourcing from the loopback make sure to advertise the PE-to-CE interface routes. You can accomplish this a few ways but for this example add protocol direct to the `vpnexport` policy on both PE1 and PE2.

SEE ALSO

Interprovider VPNs | 447

Example: Configuring Interprovider Layer 3 VPN Option C

IN THIS SECTION

Interprovider Layer 3 VPN Option C provides interprovider multihop EBGP redistribution of labeled VPN-IPv4 routes between source and destination ASs, with EBGP redistribution of labeled IPv4 routes from AS to neighboring AS. Compared to Option A and Option B, Option C is the most scalable solution. To configure an interprovider Layer 3 VPN option C service, you need to configure the AS border routers and the PE routers connected to the end customer's CE routers using multihop EBGP.

This example provides a step-by-step procedure to configure interprovider layer 3 VPN option C, which is one of the recommended implementations of MPLS VPN when that service is required by a customer that has more than one AS but not all of the customer's ASs can be serviced by the same service provider (SP). It is organized in the following sections:

Requirements

This example requires the following hardware and software components:

- Junos OS Release 9.5 or later.
- Eight Juniper Networks M Series Multiservice Edge Routers, T Series Core Routers, TX Matrix Routers, or MX Series 5G Universal Routing Platforms.

Configuration Overview and Topology

IN THIS SECTION

Interprovider layer 3 VPN option C is a very scalable interprovider VPN solution to the problem of providing VPN services to a customer that has different sites, not all of which can use the same SP.

RFC 4364 section 10, refers to this method as multihop EBGP redistribution of labeled VPN-IPv4 routes between source and destination ASs, with EBGP redistribution of labeled IPv4 routes from AS to neighboring AS.

This solution is similar to the solution described in *Implementing Interprovider Layer 3 VPN Option B*, except internal IPv4 unicast routes are advertised instead of external VPN-IPv4-unicast routes, using EBGP. Internal routes are internal to leaf SPs (SP1 and SP2 in this example), and external routes are those learned from the end customer requesting VPN services.

In this configuration:

- After the loopback address of Router PE2 is learned by Router PE1 and the loopback address of Router PE1 is learned by Router PE2, the end PE routers establish an MP-EBGP session for exchanging VPN-IPv4 routes.
- Since VPN-IPv4 routes are exchanged among end PE routers, any other router on the path from Router PE1 and Router PE2 does not need to keep or install VPN-IPv4 routes in their routing information base (RIB) or forwarding information base (FIB) tables.
- An MPLS path needs to be established between Router PE1 and Router PE2.

RFC 4364 describes only one solution that uses a BGP labeled-unicast approach. In this approach, the ASBR routers advertise the loopback addresses of the PE routers and associate each prefix with a label according to *RFC 3107*. Service providers may use RSVP or LDP to establish an LSP between ASBR routers and PE routers in their internal network.

In this network, ASBR1 receives label information associated with the loopback IP address of Router PE1 and advertises another label to Router ASBR2 using MP-EBGP labeled-unicast. Meanwhile, the ASBRs build their own MPLS forwarding table according to the received and advertised routes and labels. Router ASBR1 uses its own IP address as the next-hop information.

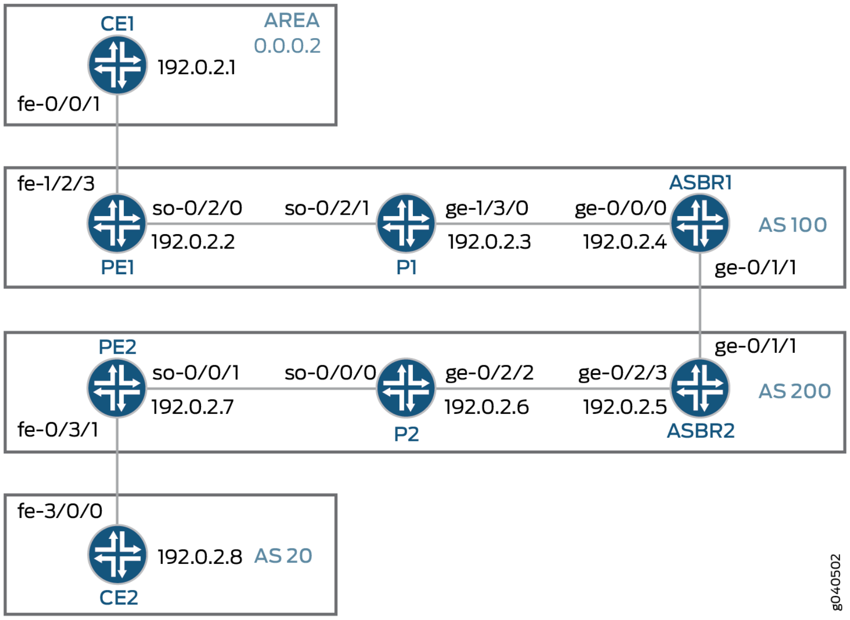
Router ASBR2 receives this prefix associated with a label, assigns another label, changes the next-hop address to its own address, and advertises it to Router PE1. Router PE1 now has an update with the label information and next-hop to Router ASBR1. Also, Router PE1 already has a label associated with the IP address of Router ASBR1. If Router PE1 sends an IP packet to Router PE2, it pushes two labels: one for the IP address of Router PE2 (obtained using MP-IBGP labeled-unicast advertisement) and one for the IP address of Router ASBR1 (obtained using LDP or RSVP).

Router ASBR1 then pops the outer label and swaps the inner label with the label learned from a neighbor ASBR for its neighboring PE router. Router ASBR2 performs a similar function and swaps the incoming label (only one) and pushes another label that is associated with the address of Router PE2. Router PE2 pops both labels and passes the remaining IP packet to its own CPU. After the end-to-end connection among the PE routers is created, the PE routers establish an MP-EBGP session to exchange VPN-IPv4 routes.

In this solution, PE routers push three labels onto the IP packet coming from the VPN end user. The inner-most label, obtained using MP-EBGP, determines the correct VPN routing and forwarding (VRF) routing instance at the remote PE. The middle label is associated with the IP address of the remote PE and is obtained from an ASBR using MP-IBGP labeled-unicast. The outer label is associated with the IP addresses of the ASBRs and is obtained using LDP or RSVP.

The physical topology of the network is shown in Figure 49 on page 500.

Figure 49: Physical Topology of Interprovider Layer 3 VPN Option C



Topology

Configuration

IN THIS SECTION

NOTE: The procedure presented here is written with the assumption that the reader is already familiar with MPLS MVPN configuration. This example focuses on explaining the unique configuration required for carrier-of-carriers solutions for VPN services to different sites.

To configure interprovider layer 3 VPN option C, perform the following tasks:

Configuring Router CE1

Step-by-Step Procedure

1. On Router CE1, configure the IP address and protocol family on the Fast Ethernet interface for the link between Router CE1 and Router PE1. Specify the `inet` address family type.

```
[edit interfaces fe-0/0/1.0]
family inet {
    address 198.51.100.1/24;
}
```

2. On Router CE1, configure the IP address and protocol family on the loopback interface. Specify the `inet` address family type.

```
[edit interfaces lo0]
unit 0 {
    family inet {
        address 192.0.2.1/32;
    }
}
```

3. On Router CE1, configure a routing protocol. The routing protocol can be a static route, RIP, OSPF, ISIS, or EBGP. In this example we configure OSPF. Include the logical interface for the link between Router CE1 and Router PE1 and the logical loopback interface of Router CE1.

```
[edit protocols]
ospf {
    area 0.0.0.2 {
        interface fe-0/0/1.0;
        interface lo0.0 {
            passive;
        }
    }
}
```

Configuring Router PE1

Step-by-Step Procedure

1. On Router PE1, configure IPv4 addresses on the SONET, Fast Ethernet, and logical loopback interfaces. Specify the `inet` address family on all of the interfaces. Specify the `mpls` address family on the SONET interfaces.

```
[edit interfaces]
so-0/2/0 {
    unit 0 {
        family inet {
            address 192.168.1.2/24;
        }
        family mpls;
    }
}
fe-1/2/3 {
    unit 0 {
        family inet {
            address 198.51.100.3/24;
        }
    }
}
lo0 {
    unit 0 {
```

```
        family inet {
            address 192.0.2.2/32;
        }

    }
}
```

2. On Router PE1, configure the routing instance for VPN2. Specify the `vrf` instance type and specify the customer-facing Fast Ethernet interface. Configure a route distinguisher to create a unique VPN-IPv4 address prefix. Apply the VRF import and export policies to enable the sending and receiving of route targets. Configure the OSPF protocol within the VRF. Specify the customer-facing Fast Ethernet interface and specify the export policy to export BGP routes into OSPF.

```
[edit routing-instances]
vpn2CE1 {
    instance-type vrf;
    interface fe-1/2/3.0;
    route-distinguisher 1:100;
    vrf-import vpnimport;
    vrf-export vpnexport;
    protocols {
        ospf {
            export bgp-to-ospf;
            area 0.0.0.2 {
                interface fe-1/2/3.0;
            }
        }
    }
}
```

3. On Router PE1, configure the RSVP and MPLS protocols to support the LSP. Configure the LSP to Router ASBR1 and specify the IP address of the logical loopback interface on Router ASBR1. Configure the OSPF protocol. Specify the core-facing SONET interface and specify the logical loopback interface on Router PE1.

```
[edit protocols]
rsvp {
    interface so-0/2/0.0;
    interface lo0.0;
}
mpls {
```

```
    label-switched-path To-ASBR1 {
        to 192.0.2.4;
    }
    interface so-0/2/0.0;
    interface lo0.0;
}
ospf {
    traffic-engineering;
    area 0.0.0.0 {
        interface so-0/2/0.0;
        interface lo0.0 {
            passive;
        }
    }
}
```

4. On Router PE1, configure the `To_ASBR1` peer BGP group. Specify the group type as `internal`. Specify the local address as the logical loopback interface on Router PE1. Specify the neighbor address as the logical loopback interface on Router ASBR1. Specify the `inet` address family. For a PE router to install a route in the VRF, the next hop must resolve to a route stored within the `inet.3` table. The `labeled-unicast resolve-vpn` statements allow labeled routes to be placed in the `inet.3` routing table for route resolution, which are then resolved for PE router connections where the remote PE is located across another AS.

```
[edit protocols]
bgp {
    group To_ASBR1 {
        type internal;
        local-address 192.0.2.2;
        neighbor 192.0.2.4 {
            family inet {
                labeled-unicast {
                    resolve-vpn;
                }
            }
        }
    }
}
```

5. On Router PE1, configure multihop EBGP toward PE2. Specify the `inet-vpn` family.

```
[edit protocols]
bgp {
    group To_PE2 {
        multihop {
            ttl 20;
        }
        local-address 192.0.2.2;
        family inet-VPN {
            unicast;
        }
        neighbor 192.0.2.7 {
            peer-as 200;
        }
    }
}
```

6. On Router PE1, configure the BGP local autonomous system number.

```
[edit routing-options]
autonomous-system 100;
```

7. On Router PE1, configure a policy to export the BGP routes into OSPF.

```
[edit policy-options]
policy-statement bgp-to-ospf {
    term 1 {
        from protocol bgp;
        then accept;
    }
    term 2 {
        then reject;
    }
}
```

8. On Router PE1, configure a policy to add the VRF route target to the routes being advertised for this VPN.

```
[edit policy-options]
    policy-statement vpnexport {
        term 1 {
            from protocol ospf;
            then {
                community add test_comm;
                accept;
            }
        }
        term 2 {
            then reject;
        }
    }
```

9. On Router PE1, configure a policy to import routes from BGP that have the `test_comm` community attached.

```
[edit policy-options]
    policy-statement vpnimport {
        term 1 {
            from {
                protocol bgp;
                community test_comm;
            }
            then accept;
        }
        term 2 {
            then reject;
        }
    }
```

10. On Router PE1, define the `test_comm` BGP community with a route target.

```
[edit policy-options]
community test_comm members target:1:100;
```

Configuring Router P1

Step-by-Step Procedure

1. On Router P1, configure IP addresses for the SONET and Gigabit Ethernet interfaces. Enable the interfaces to process the `inet` and `mpls` address families. Configure the IP address for the `lo0.0` loopback interface and enable the interface to process the `inet` address family.

   ```
   [edit interfaces]
   so-0/2/1 {
       unit 0 {
           family inet {
               address 192.168.1.4/24;
           }
           family mpls;
       }
   }
   ge-1/3/0 {
       unit 0 {
           family inet {
               address 192.168.2.5/24;
           }
           family mpls;
       }
   }
   lo0 {
       unit 0 {
           family inet {
               address 192.0.2.3/32;
           }
       }
   }
   ```

2. On Router P1, configure the RSVP and MPLS protocols to support the LSP. Specify the SONET and Gigabit Ethernet interfaces.

   Configure the OSPF protocol. Specify the SONET and Gigabit Ethernet interfaces and specify the logical loopback interface. Enable OSPF to support traffic engineering extensions.

   ```
   [edit protocols]
   rsvp {
   ```

```
    interface so-0/2/1.0;
    interface ge-1/3/0.0;
    interface lo0.0;
}
mpls {
    interface lo0.0;
    interface ge-1/3/0.0;
    interface so-0/2/1.0;
}
ospf {
    traffic-engineering;
    area 0.0.0.0 {
        interface ge-1/3/0.0;
        interface so-0/2/1.0;
        interface lo0.0 {
            passive;
        }
    }
}
```

Configuring Router ASBR1

Step-by-Step Procedure

1. On Router ASBR1, configure IP addresses for the Gigabit Ethernet interfaces. Enable the interfaces to process the `inet` and `mpls` addresses families. Configure the IP addresses for the `lo0.0` loopback interface and enable the interface to process the `inet` address family.

```
[edit interfaces]
ge-0/0/0 {
    unit 0 {
        family inet {
            address 192.168.2.6/24;
        }
        family mpls;
    }
}
ge-0/1/1 {
    unit 0 {
        family inet {
            address 192.168.3.7/24;
```

```
        }
        family mpls;
    }
}
lo0 {
    unit 0 {
        family inet {
            address 192.0.2.4/32;
        }
    }
}
```

2. On Router ASBR1, configure the protocols to support the LSP.

   Configure the RSVP protocol by specifying the Gigabit Ethernet interface that is facing the P1 router and the logical loopback interface.

   Configure the MPLS protocol by specifying the Gigabit Ethernet interfaces and the logical loopback interface. Include the `traffic-engineering bgp-igp-both-ribs` statement at the `[edit protocols mpls]` hierarchy level.

   Configure the OSPF protocol on the Gigabit Ethernet interface facing the P1 router and the logical loopback interface. Enable OSPF to support traffic engineering extensions.

```
[edit protocols]
rsvp {
    interface ge-0/0/0.0;
    interface lo0.0;
}
mpls {
    traffic-engineering bgp-igp-both-ribs;
    label-switched-path To_PE1 {
        to 192.0.2.2;
    }
    interface lo0.0;
    interface ge-0/0/0.0;
    interface ge-0/1/1.0;
}
ospf {
    traffic-engineering;
    area 0.0.0.0 {
        interface ge-0/0/0.0;
        interface lo0.0 {
```

```
            passive;
        }
    }
}
```

3. On Router ASBR1, create the `To-PE1` internal BGP peer group. Specify the local IP peer address as the local `lo0.0` address. Specify the neighbor IP peer address as the Gigabit Ethernet interface address of Router PE1.

```
[edit protocols]
bgp {
    group To-PE1 {
        type internal;
        local-address 192.0.2.4;
        neighbor 192.0.2.2 {
            family inet {
                labeled-unicast;
            }
            export next-hop-self;
        }
    }
```

4. On Router ASBR1, create the `To-ASBR2` external BGP peer group. Enable the router to use BGP to advertise network layer reachability information (NLRI) for unicast routes. Specify the neighbor IP peer address as the Gigabit Ethernet interface address on Router ASBR2.

```
[edit protocols]
group To-ASBR2 {
    type external;
    family inet {
        labeled-unicast;
    }
    export To-ASBR2;
    neighbor 192.168.3.8 {
        peer-as 200;
    }
}
```

5. On Router ASBR1, configure the BGP local autonomous system number.

```
[edit routing-options]
autonomous-system 100;
```

6. On Router ASBR 1, configure a policy to import routes from BGP that match the 192.0.2.2/24 route.

```
[edit policy-options]
policy-statement To-ASBR2 {
    term 1 {
        from {
            route-filter 192.0.2.2/32 exact;
        }
        then accept;
    }
    term 2 {
        then reject;
    }
```

7. On Router ASBR 1, define a next-hop self policy and apply it to the IBGP sessions.

```
[edit policy-options]
policy-statement next-hop-self {
    then {
        next-hop self;
    }
}
```

Configuring Router ASBR2

Step-by-Step Procedure

1. On Router ASBR2, configure IP addresses for the Gigabit Ethernet interfaces. Enable the interfaces to process the `inet` and `mpls` address families. Configure the IP address for the `lo0.0` loopback interface and enable the interface to process the `inet` address family.

```
[edit interfaces]
ge-0/1/1 {
```

```
    unit 0 {
        family inet {
            address 192.168.3.8/24;
        }
        family mpls;
    }
}
ge-0/2/3 {
    unit 0 {
        family inet {
            address 192.168.4.9/24;
        }
        family mpls;
    }
}
lo0 {
    unit 0 {
        family inet {
            address 192.0.2.5/32;
        }
    }
}
```

2. On Router ASBR2, configure the protocols to support the LSP.

   Configure the RSVP protocol by specifying the Gigabit Ethernet interface facing the P2 router and the logical loopback interface .

   Configure the MPLS protocol by specifying the Gigabit Ethernet interfaces and the logical loopback interface. Include the `traffic-engineering bgp-igp-both-ribs` statement at the `[edit protocols mpls]` hierarchy level.

   Configure the OSPF protocol on the Gigabit Ethernet interface facing the P2 router and the logical loopback interface . Enable OSPF to support traffic engineering extensions.

```
[edit protocols]
rsvp {
    interface ge-0/2/3.0;
    interface lo0.0;
}
mpls {
    traffic-engineering bgp-igp-both-ribs;
    label-switched-path To_PE2 {
```

```
        to 192.0.2.7;
    }
    interface lo0.0
    interface ge-0/2/3.0;
    interface ge-0/1/1.0;
}
ospf {
    traffic-engineering;
    area 0.0.0.0 {
        interface ge-0/2/3.0;
        interface lo0.0 {
            passive;
        }
    }
}
```

3. On Router ASBR2, create the `To-PE2` internal BGP peer group. Specify the local IP peer address as the local `lo0.0` address. Specify the neighbor IP peer address as the `lo0.0` interface address of Router PE2.

```
[edit protocols]
bgp {
    group To-PE2 {
        type internal;
        local-address 192.0.2.5;
        export next-hop-self;
        neighbor 192.0.2.7 {
            family inet {
                labeled-unicast;
            }
            export next-hop-self;
        }
    }
}
```

4. On Router ASBR2, create the `To-ASBR1` external BGP peer group. Enable the router to use BGP to advertise NLRI for unicast routes. Specify the neighbor IP peer address as the Gigabit Ethernet interface address on Router ASBR1.

```
[edit protocols]
bgp {
    group To-ASBR1 {
```

```
        type external;
        family inet {
            labeled-unicast;
        }
        export To-ASBR1;
        neighbor 192.168.3.7 {
            peer-as 100;
        }
    }
}
```

5. On Router ASBR2 configure the BGP local autonomous system number.

```
[edit routing-options]
autonomous-system 200;
```

6. On Router ASBR2, configure a policy to import routes from BGP that match the `192.0.2.7/24` route.

```
[edit policy-options]
policy-statement To-ASBR1 {
    term 1 {
        from {
            route-filter 192.0.2.7/32 exact;
        }
        then accept;
    }
    term 2 {
        then reject;
    }
}
```

7. On Router ASBR 2, define a next-hop self policy.

```
[edit policy-options]
policy-statement next-hop-self {
    then {
        next-hop self;
    }
}
```

Configuring Router P2

Step-by-Step Procedure

1. On Router P2, configure IP addresses for the SONET and Gigabit Ethernet interfaces. Enable the interfaces to process the `inet` and `mpls` addresses families. Configure the IP addresses for the `lo0.0` loopback interface and enable the interface to process the `inet` address family.

```
[edit interfaces]
so-0/0/0 {
    unit 0 {
        family inet {
            address 192.168.5.10/24;
        }
        family mpls;
    }
}
ge-0/2/2 {
    unit 0 {
        family inet {
            address 192.168.4.11/24;
        }
        family mpls;
    }
}
lo0 {
    unit 0 {
        family inet {
            address 192.0.2.6/32;
        }
    }
}
```

2. On Router P2, configure the RSVP and MPLS protocols to support the LSP. Specify the SONET and Gigabit Ethernet interfaces.

   Configure the OSPF protocol. Specify the SONET and Gigabit Ethernet interfaces and specify the logical loopback interface. Enable OSPF to support traffic engineering extensions.

```
[edit protocols]
rsvp {
```

```
    interface so-0/0/0.0;
    interface ge-0/2/2.0;
    interface lo0.0;
}
mpls {
    interface lo0.0;
    interface ge-0/2/2.0;
    interface so-0/0/0.0;
}
ospf {
    traffic-engineering;
    area 0.0.0.0 {
        interface ge-0/2/2.0;
        interface so-0/0/0.0;
        interface lo0.0 {
            passive;
        }
    }
}
```

Configuring Router PE2

Step-by-Step Procedure

1. On Router PE2, configure IPv4 addresses on the SONET, Fast Ethernet, and logical loopback interfaces. Specify the `inet` address family on all of the interfaces. Specify the `mpls` address family on the SONET interface.

```
[edit interfaces]
so-0/0/1 {
    unit 0 {
        family inet {
            address 192.168.5.12/24;
        }
        family mpls;
    }
}
fe-0/3/1 {
    unit 0 {
        family inet {
            address 192.168.6.13/24;
```

```
        }
    }
}
lo0 {
    unit 0 {
        family inet {
            address 192.0.2.7/32;
        }
    }
}
```

2. On Router PE2, configure the routing instance for VPN2. Specify the `vrf` instance type and specify the customer-facing Fast Ethernet interface. Configure a route distinguisher to create a unique VPN-IPv4 address prefix. Apply the VRF import and export policies to enable the sending and receiving of route targets. Configure the BGP peer group within the VRF. Specify AS `20` as the peer AS and specify the IP address of the Fast Ethernet interface on Router CE1 as the neighbor address.

```
[edit routing-instances]
vpn2CE2 {
    instance-type vrf;
    interface fe-0/3/1.0;
    route-distinguisher 1:100;
    vrf-import vpnimport;
    vrf-export vpnexport;
    protocols {
        bgp {
            group To_CE2 {
                peer-as 20;
                neighbor 192.168.6.14;
            }
        }
    }
}
```

3. On Router PE2, configure the RSVP and MPLS protocols to support the LSP. Configure the LSP to ASBR2 and specify the IP address of the logical loopback interface on Router ASBR2. Configure the OSPF protocol. Specify the core-facing SONET interface and specify the logical loopback interface on Router PE2.

```
[edit protocols]
rsvp {
```

```
    interface so-0/0/1.0;
    interface lo0.0;
}
mpls {
    label-switched-path To-ASBR2 {
        to 192.0.2.5;
    }
    interface so-0/0/1.0;
    interface lo0.0;
}
ospf {
    traffic-engineering;
    area 0.0.0.0 {
        interface so-0/0/1.0;
        interface lo0.0 {
            passive;
        }
    }
}
```

4. On Router PE2, configure the `To_ASBR2` BGP group. Specify the group type as `internal`. Specify the local address as the logical loopback interface on Router PE2. Specify the neighbor address as the logical loopback interface on the Router ASBR2.

```
[edit protocols]
bgp {
    group To_ASBR2 {
        type internal;
        local-address 192.0.2.7;
        neighbor 192.0.2.5 {
            family inet {
                labeled-unicast {
                    resolve-vpn;
                }
            }
        }
    }
}
```

5. On Router PE2, configure multihop EBGP towards Router PE1 Specify the `inet-vpn` address family.

```
[edit protocols]
bgp {
    group To_PE1 {
        type external;
        local-address 192.0.2.7;
        multihop {
            ttl 20;
        }
        family inet-vpn {
            unicast;
        }
        neighbor 192.0.2.2 {
            peer-as 100;
        }
    }
}
```

6. On Router PE2, configure the BGP local autonomous system number.

```
[edit routing-options]
autonomous-system 200;
```

7. On Router PE2, configure a policy to add the VRF route target to the routes being advertised for this VPN.

```
[edit policy-options]
policy-statement vpnexport {
    term 1 {
        from protocol bgp;
        then {
            community add test_comm;
            accept;
        }
    }
    term 2 {
        then reject;
```

```
    }
}
```

8. On Router PE2, configure a policy to import routes from BGP that have the `test_comm` community attached.

```
[edit policy-options]
policy-statement vpnimport {
    term 1 {
        from {
            protocol bgp;
            community test_comm;
        }
        then accept;
    }
    term 2 {
        then reject;
    }
}
```

9. On Router PE1, define the `test_comm` BGP community with a route target.

```
[edit policy-options]
community test_comm members target:1:100;
```

Configuring Router CE2

Step-by-Step Procedure

1. On Router CE2, configure the IP address and protocol family on the Fast Ethernet interface for the link between Router CE2 and Router PE2. Specify the `inet` address family type.

```
[edit interfaces]
fe-3/0/0 {
    unit 0 {
        family inet {
            address 192.168.6.14/24;
        }
```

```
    }
}
```

2. On Router CE2, configure the IP address and protocol family on the loopback interface. Specify the `inet` address family type.

```
[edit interfaces lo0]
lo0 {
    unit 0 {
        family inet {
            address 192.0.2.8/32;
        }
    }
}
```

3. On Router CE2, define a policy named `myroutes` that accepts direct routes.

```
[edit policy-options]
policy-statement myroutes {
    from protocol direct;
    then accept;
}
```

4. On Router CE2, configure a routing protocol. The routing protocol can be a static route, RIP, OSPF, ISIS, or EBGP. In this example, we configure EBGP. Specify the BGP neighbor IP address as the logical loopback interface of Router PE1. Apply the `myroutes` policy.

```
[edit protocols]
bgp {
    group To_PE2 {
        neighbor 198.51.100.13 {
            export myroutes;
            peer-as 200;
        }
    }
}
```

5. On Router CE2, configure the BGP local autonomous system number.

```
[edit routing-options]
autonomous-system 20;
```

Verifying the VPN Operation

Step-by-Step Procedure

1. Commit the configuration on each router.

   NOTE: The MPLS labels shown in this example will be different than the labels used in your configuration.

2. On Router PE1, display the routes for the `vpn2CE1` routing instance using the `show ospf route` command. Verify that the `192.0.2.1` route is learned from OSPF.

```
user@PE1> show ospf route instance vpn2CE1
Topology default Route Table:

Prefix             Path   Route       NH   Metric  NextHop       Nexthop
                   Type   Type        Type         Interface     addr/label
192.0.2.1            Intra  Router      IP        1  fe-1/2/3.0    198.51.100.1
192.0.2.1/32          Intra  Network     IP        1  fe-1/2/3.0    198.51.100.1
198.51.100.0/24      Intra Network    IP            1 fe-1/2/3.0
```

3. On Router PE1, use the `show route advertising-protocol` command to verify that Router PE1 advertises the `192.0.2.1` route to Router PE2 using MP-BGP with the VPN MPLS label.

```
user@PE1> show route advertising-protocol bgp 192.0.2.7 extensive
bgp.l3vpn.0: 2 destinations, 2 routes (2 active, 0 holddown, 0 hidden)
* 1:100:192.0.2.1/32 (1 entry, 1 announced)
 BGP group To_PE2 type External
     Route Distinguisher: 1:100
     VPN Label: 300016
     Nexthop: Self
     Flags: Nexthop Change
     MED: 1
```

```
     AS path: [100] I
     Communities: target:1:100 rte-type:0.0.0.2:1:0
```

4. On Router ASBR1, use the `show route advertising-protocol` command to verify that Router ASBR1 advertises the `192.0.2.2` route to Router ASBR2.

```
user@ASBR1> show route advertising-protocol bgp 192.168.3,8 extensive
inet.0: 14 destinations, 16 routes (14 active, 0 holddown, 0 hidden)
* 192.0.2.2/32 (2 entries, 1 announced)
 BGP group To-PE2 type External
     Route Label: 300172
     Nexthop: Self
     Flags: Nexthop Change
     MED: 2
     AS path: [100] I
```

5. On Router ASBR2, use the `show route receive-protocol` command to verify that the router receives and accepts the `192.0.2.2` route .

```
user@ASBR2> show route receive-protocol bgp 192.168.3.7 extensive
inet.0: 10 destinations, 11 routes (10 active, 0 holddown, 0 hidden)
* 192.0.2.2/32 (1 entry, 1 announced)
     Accepted
     Route Label: 300172
     Nexthop: 192.168.3.7
     MED: 2
     AS path: 100 I
```

6. On Router ASBR2, use the `show route advertising-protocol` command to verify that Router ASBR2 advertises the `192.0.2.2` route to Router PE2.

```
user@ASBR2> show route advertising-protocol bgp 192.0.2.7 extensive
inet.0: 10 destinations, 11 routes (10 active, 0 holddown, 0 hidden)
* 192.0.2.2/32 (1 entry, 1 announced)
 BGP group To-PE2 type Internal
     Route Label: 300192
     Nexthop: Self
     Flags: Nexthop Change
     MED: 2
```

```
     Localpref: 100
     AS path: [200] 100 I
```

7. On Router PE2, use the `show route receive-protocol` command to verify that Router PE2 receives the route and puts it in the `inet.0.` routing table. Verify that Router PE2 also receives the update from Router PE1 and accepts the route.

```
user@PE2> show route receive-protocol bgp 192.0.2.5 extensive
inet.0: 13 destinations, 14 routes (13 active, 0 holddown, 0 hidden)
* 192.0.2.2/32 (1 entry, 1 announced)
     Accepted
     Route Label: 300192
     Nexthop: 192.0.2.5
     MED: 2
     Localpref: 100
     AS path: 100 I
     AS path: Recorded

inet.3: 2 destinations, 2 routes (2 active, 0 holddown, 0 hidden)

* 192.0.2.2/32  (1 entry, 1 announced)
     Accepted
     Route Label: 300192
     Nexthop: 192.0.2.5
     MED: 2
     Localpref: 100
     AS path: 100 I
     AS path: Recorded
```

8. On Router PE2, use the `show route receive-protocol` command to verify that Router PE2 puts the route in the routing table of the `vpn2CE2` routing instance and advertises the route to Router CE2 using EBGP.

```
user@PE2> show route receive-protocol bgp 192.0.2.2 detail
inet.0: 17 destinations, 18 routes (17 active, 0 holddown, 0 hidden)

inet.3: 6 destinations, 6 routes (6 active, 0 holddown, 0 hidden)

__juniper_private1__.inet.0: 14 destinations, 14 routes (8 active, 0 holddown, 6 hidden)

__juniper_private2__.inet.0: 1 destinations, 1 routes (0 active, 0 holddown, 1 hidden)
```

```
vpn2CE2.inet.0: 4 destinations, 5 routes (4 active, 0 holddown, 0 hidden)
* 192.0.2.1/32 (1 entry, 1 announced)
     Accepted
     Route Distinguisher: 1:100
     VPN Label: 300016
     Nexthop: 192.0.2.2
     MED: 1
     AS path: 100 I
     AS path: Recorded
     Communities: target:1:100 rte-type:0.0.0.2:1:0

iso.0: 1 destinations, 1 routes (1 active, 0 holddown, 0 hidden)

mpls.0: 4 destinations, 4 routes (4 active, 0 holddown, 0 hidden)

bgp.l3vpn.0: 2 destinations, 2 routes (2 active, 0 holddown, 0 hidden)

* 1:100:192.0.2.1/32 (1 entry, 0 announced)
     Accepted
     Route Distinguisher: 1:100
     VPN Label: 300016
     Nexthop: 192.0.2.2
     MED: 1
     AS path: 100 I
     AS path: Recorded
     Communities: target:1:100 rte-type:0.0.0.2:1:0

__juniper_private1__.inet6.0: 4 destinations, 4 routes (4 active, 0 holddown, 0 hidden)
```

9. On Router PE2, use the `show route advertising-protocol` command to verify that Router PE2 advertises the `192.0.2.1` route to Router CE2 through the `vpn2CE2` peer group.

```
user@PE2> show route advertising-protocol bgp 192.168.6.14 extensive
vpn2CE2.inet.0: 4 destinations, 5 routes (4 active, 0 holddown, 0 hidden)
* 192.0.2.1/32 (1 entry, 1 announced)
 BGP group vpn2CE2 type External
     Nexthop: Self
     AS path: [200] 100 I
     Communities: target:1:100 rte-type:0.0.0.2:1:0
```

10. On Router CE2, use the `show route` command to verify that Router CE2 receives the `192.0.2.1` route from Router PE2.

```
user@CE2> show route 192.0.2.1
inet.0: 6 destinations, 6 routes (6 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

192.0.2.1/32         *[BGP/170] 00:25:36, localpref 100
                      AS path: 200 100 I
                     > to 192.168.6.13 via fe-3/0/0.0
```

11. On Router CE2, use the `ping` command and specify `192.0.2.8` as the source of the ping packets to verify connectivity with Router CE1.

```
user@CE2> ping 192.0.2.1 source 192.0.2.8
PING 192.0.2.1 (192.0.2.1): 56 data bytes
64 bytes from 192.0.2.1: icmp_seq=0 ttl=58 time=4.786 ms
64 bytes from 192.0.2.1: icmp_seq=1 ttl=58 time=10.210 ms
64 bytes from 192.0.2.1: icmp_seq=2 ttl=58 time=10.588 ms
```

12. On Router PE2, use the `show route` command to verify that the traffic is sent with an inner label of `300016`, a middle label of `300192`, and a top label of `299776`.

```
user@PE2> show route 192.0.2.1 detail
vpn2CE2.inet.0: 4 destinations, 5 routes (4 active, 0 holddown, 0 hidden)
192.0.2.1/32 (1 entry, 1 announced)
        *BGP    Preference: 170/-101
                Route Distinguisher: 1:100
                Next hop type: Indirect
                Next-hop reference count: 3
                Source: 192.0.2.2
                Next hop type: Router, Next hop index: 653
                Next hop: via so-0/0/1.0 weight 0x1, selected
                Label-switched-path To-ASBR2
             Label operation: Push 300016, Push 300192, Push 299776(top)
                Protocol next hop: 192.0.2.2
                Push 300016
                Indirect next hop: 8c61138 262142
                State: <Secondary Active Ext>
                Local AS:   200 Peer AS:   100
```

```
                Age: 17:33      Metric: 1       Metric2: 2
                Task: BGP_100.192.0.2.2+62319
                Announcement bits (3): 0-RT 1-KRT 2-BGP RT Background
                AS path: 100 I
                AS path: Recorded
                Communities: target:1:100 rte-type:0.0.0.2:1:0
                Accepted
                VPN Label: 300016
                Localpref: 100
                Router ID: 192.0.2.2
                Primary Routing Table bgp.l3vpn.0
```

13. On Router ASBR2, use the `show route table` command to verify that Router ASBR2 receives the traffic after the top label is popped by Router P2. Verify that label `300192` is a swapped with label `300176` and the traffic is sent towards Router ASBR1 using interface ge-0/1/1.0. At this point, the bottom label `300016` is preserved.

```
user@ASBR2# show route table mpls.0 detail
300192 (1 entry, 1 announced)
        *VPN    Preference: 170
                Next hop type: Router, Next hop index: 660
                Next-hop reference count: 2
                Source: 192.168.3.7                Next hop: 192.168.3.7 via ge-0/1/1.0,
selected
                Label operation: Swap 300176
                State: <Active Int Ext>
                Local AS:   200
                Age: 24:01
                Task: BGP RT Background
                Announcement bits (1): 0-KRT
                AS path: 100 I
                Ref Cnt: 1
```

14. On Router ASBR1, use the `show route table` command to verify that when Router ASBR1 receives traffic with label `300176`, it swaps the label with `299824` to reach Router PE1.

```
user@ASBR1> show route table mpls.0 detail
300176 (1 entry, 1 announced)
        *VPN    Preference: 170
                Next hop type: Router, Next hop index: 651
                Next-hop reference count: 2
```

```
                Next hop: 192.168.2.5 via ge-0/0/0.0 weight 0x1, selected
                Label operation: Swap 299824
                State: <Active Int Ext>
                Local AS:   100
                Age: 25:53
                Task: BGP RT Background
                Announcement bits (1): 0-KRT
                AS path: I
                Ref Cnt: 1
```

15. On Router PE1, use the `show route table` command to verify that Router PE1 receives the traffic after the top label is popped by Router P1. Verify that label `300016` is popped and the traffic is sent towards Router CE1 using interface `fe-1/2/3.0`.

```
user@PE1>  show route table mpls.0 detail
300016 (1 entry, 1 announced)
        *VPN    Preference: 170
                Next hop type: Router, Next hop index: 643
                Next-hop reference count: 2
                Next hop: 198.51.100.1 via fe-1/2/3.0, selected
                Label operation: Pop
                State:< Active Int Ext>
                Local AS:   100
                Age: 27:37
                Task: BGP RT Background
                Announcement bits (1): 0-KRT
                AS path: I
                Ref Cnt: 1
                Communities: rte-type:0.0.0.2:1:0
```

SEE ALSO

Interprovider VPNs | 447

Carrier-of-Carrier VPNs

IN THIS SECTION

Understanding Carrier-of-Carriers VPNs

IN THIS SECTION

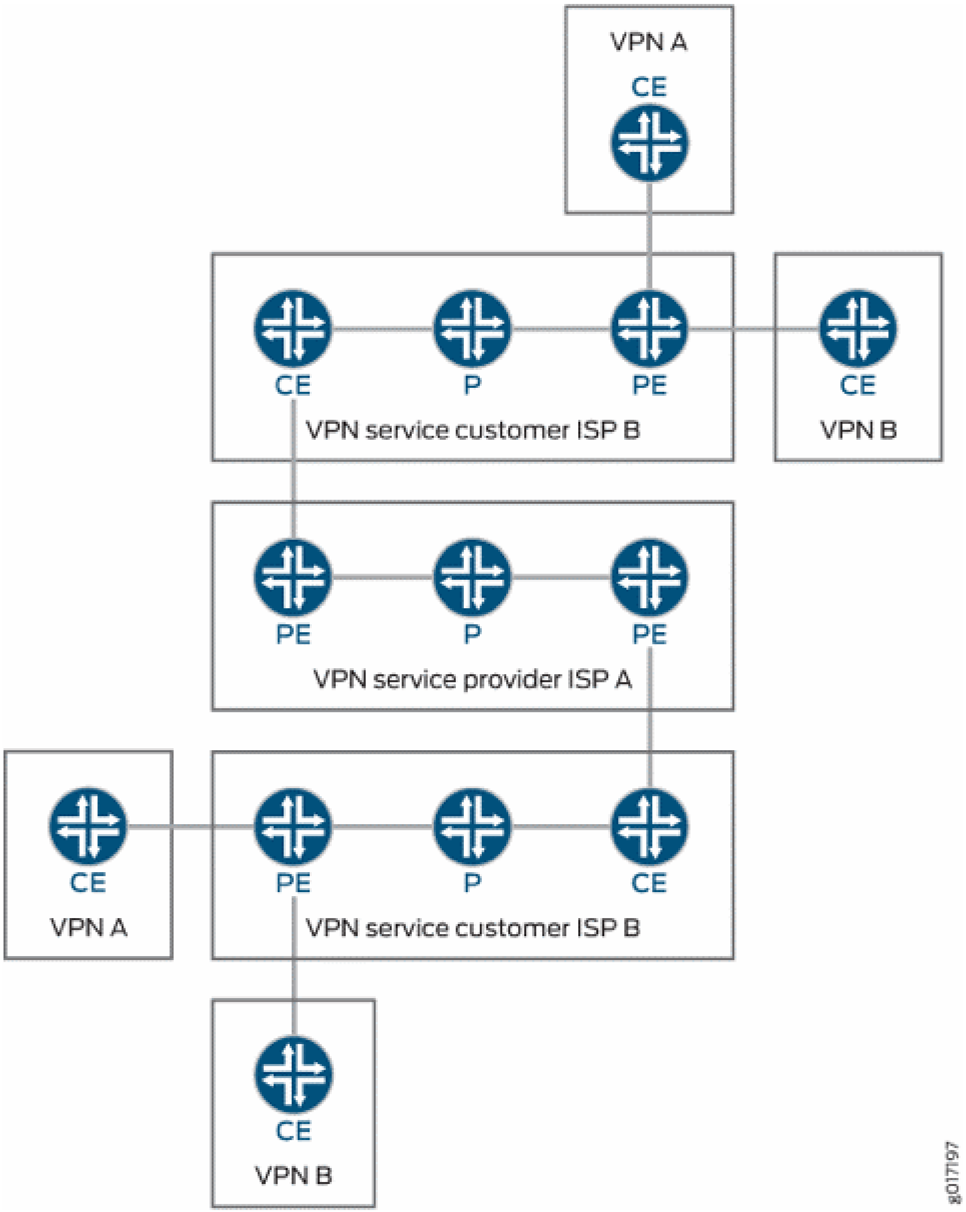
The customer of a VPN service provider might be a service provider for the end customer. The following are the two main types of carrier-of-carriers VPNs (as described in RFC 4364:

- "Internet Service Provider as the Customer" on page 530—The VPN customer is an ISP that uses the VPN service provider’s network to connect its geographically disparate regional networks. The customer does not have to configure MPLS within its regional networks.

- "VPN Service Provider as the Customer" on page 531—The VPN customer is itself a VPN service provider offering VPN service to its customers. The carrier-of-carriers VPN service customer relies on the backbone VPN service provider for inter-site connectivity. The customer VPN service provider is required to run MPLS within its regional networks.

Figure 50 on page 530 illustrates the network architecture used for a carrier-of-carriers VPN service.

Figure 50: Carrier-of-Carriers VPN Architecture



This topic covers the following:

Internet Service Provider as the Customer

In this type of carrier-of-carriers VPN configuration, ISP A configures its network to provide Internet service to ISP B. ISP B provides the connection to the customer wanting Internet service, but the actual Internet service is provided by ISP A.

This type of carrier-of-carriers VPN configuration has the following characteristics:

- The carrier-of-carriers VPN service customer (ISP B) does not need to configure MPLS on its network.
- The carrier-of-carriers VPN service provider (ISP A) must configure MPLS on its network.
- MPLS must also be configured on the CE routers and PE routers connected together in the carrier-of-carriers VPN service customer’s and carrier-of-carriers VPN service provider’s networks.

VPN Service Provider as the Customer

A VPN service provider can have customers that are themselves VPN service providers. In this type of configuration, also called a hierarchical or recursive VPN, the customer VPN service provider’s VPN-IPv4 routes are considered external routes, and the backbone VPN service provider does not import them into its VRF table. The backbone VPN service provider imports only the customer VPN service provider’s internal routes into its VRF table.

The similarities and differences between interprovider and carrier-of-carriers VPNs are shown in Table 7 on page 531.

Table 7: Comparison of Interprovider and Carrier-of-Carriers VPNs

Feature	ISP Customer	VPN Service Provider Customer
Customer edge device	AS border router	PE router
IBGP sessions	Carry IPv4 routes	Carry external VPN-IPv4 routes with associated labels
Forwarding within the customer network	MPLS is optional	MPLS is required

Support for VPN service as the customer is supported on QFX10000 switches starting with Junos OS Release 17.1R1.

Configuring Carrier-of-Carriers VPNs for Customers That Provide Internet Service

IN THIS SECTION

You can configure a carrier-of-carriers VPN service for customers who want to provide basic Internet service. The carrier-of-carriers VPN service provider must configure MPLS in its network, although this configuration is optional for the carrier service customer. "Carrier-of-Carriers VPN Architecture" on page 529 shows how the routers or switches in this type of service interconnect.

To configure a carrier-of-carriers VPN, perform the tasks described in the following sections:

Configuring the Carrier-of-Carriers VPN Service Customer's CE Router

The carrier-of-carriers VPN service customer's router (or switch) acts as a CE router with respect to the service provider's PE router or switch. The following sections describe how to configure the carrier-of-carriers VPN service customer's CE router or switch:

Configuring MPLS

To configure MPLS on the customer's CE router or switch, include the `mpls` statement:

```
mpls {
    traffic-engineering bgp-igp;
    interface interface-name;
}
```

You can include this statement at the following hierarchy levels:

- `[edit protocols]`
- `[edit logical-systems logical-system-name protocols]`

Configuring BGP

To configure a group to collate the customer's internal routes, include the `bgp` statement:

```
bgp {
    group group-name {
        type internal;
        local-address address;
        neighbor address;
    }
}
```

You can include this statement at the following hierarchy levels:

- `[edit protocols]`
- `[edit logical-systems logical-system-name protocols]`

The customer’s CE router (or switch) must be able to send labels to the VPN service provider’s router. Enable this by including the `labeled-unicast` statement in the configuration for the BGP group:

```
bgp {
    group group-name {
        export internal;
        peer-as as-number;
        neighbor address {
            family inet {
                labeled-unicast;
            }
        }
    }
}
```

You can include the `bgp` statement at the following hierarchy levels:

- `[edit protocols]`
- `[edit logical-systems logical-system-name protocols]`

Configuring OSPF

To configure OSPF on the customer’s CE router or switch, include the `ospf` statement:

```
ospf {
    area area-id {
        interface interface-name {
            passive;
        }
        interface interface-name;
    }
}
```

You can include this statement at the following hierarchy levels:

- `[edit protocols]`
- `[edit logical-systems logical-system-name protocols]`

Configuring Policy Options

To configure policy options on the customer’s CE router or switch, include the `policy-statement` statement:

```
policy-statement statement-name {
    term term-name {
        from protocol [ospf direct ldp];
        then accept;
    }
    term term-name {
        then reject;
    }
}
```

You can include this statement at the following hierarchy levels:

- `[edit policy-options]`
- `[edit logical-systems logical-system-name policy-options]`

Configuring the Carrier-of-Carriers VPN Service Provider’s PE Routers

The service provider’s PE routers connect to the customer’s CE routers and forward the customer’s VPN traffic across the provider’s network.

The following sections describe how to configure the carrier-of-carriers VPN service provider’s PE routers:

Configuring MPLS

To configure MPLS on the provider’s PE routers or switches include the `mpls` statement:

```
mpls {
    interface interface-name;
    interface interface-name;
}
```

You can include this statement at the following hierarchy levels:

- `[edit protocols]`
- `[edit logical-systems logical-system-name protocols]`

Configuring BGP

To configure a BGP session with the provider PE router at the other end of the provider's network, include the `bgp` statement:

```
bgp {
    group group-name {
        type internal;
        local-address address;
        family inet-vpn {
            any;
        }
        neighbor address;
    }
}
```

You can include this statement at the following hierarchy levels:

- `[edit protocols]`
- `[edit logical-systems logical-system-name protocols]`

Configuring IS-IS

To configure IS-IS on the provider's PE routers or switches, include the `isis` statement:

```
isis {
    interface interface-name;
    interface interface-name {
        passive;
    }
}
```

You can include this statement at the following hierarchy levels:

- `[edit protocols]`
- `[edit logical-systems logical-system-name protocols]`

Configuring LDP

To configure LDP on the provider's PE routers or switches, include the `ldp` statement:

```
ldp {
    interface interface-name;
}
```

You can include this statement at the following hierarchy levels:

- `[edit protocols]`
- `[edit logical-systems logical-system-name protocols]`

Configuring a Routing Instance

To configure Layer 3 VPN service with the customer's CE router or switch, include the `labeled-unicast` statement in the configuration for the routing instance so the PE router (or switch) can send labels to the customer's CE router or switch:

```
routing-instance-name {
    instance-type vrf;
    interface interface-name;
    route-distinguisher address;
    vrf-import policy-name;
    vrf-export policy-name;
    protocols {
        bgp {
            group group-name {
                peer-as as-number;
                neighbor address {
                    family inet {
                        labeled-unicast;
                    }
                }
            }
        }
    }
}
```

You can include these statements at the following hierarchy levels:

- `[edit routing-instances]`
- `[edit logical-systems logical-system-name routing-instances]`

Configuring Policy Options

To configure a policy statement to import routes from the customer's CE router or switch, include the `policy-statement` statement:

```
policy-statement policy-name {
    term term-name {
        from {
            protocol bgp;
            community community-name;
        }
        then accept;
    }
    term term-name {
        then reject;
    }
}
```

You can include this statement at the following hierarchy levels:

- `[edit policy-options]`
- `[edit logical-systems logical-system-name policy-options]`

To configure a policy statement to export routes to the customer's CE router or switch, include the `policy-statement` and `community` statements:

```
policy-statement policy-name {
    term term-name {
        from protocol bgp;
        then {
            community add community-name;
            accept;
        }
    }
    term term-name {
        then reject;
    }
```

```
}
community community-name members value;
```

You can include these statements at the following hierarchy levels:

- `[edit policy-options]`
- `[edit logical-systems logical-system-name policy-options]`

SEE ALSO

> MPLS Feature Support on QFX Series and EX4600 Switches

Carrier-of-Carriers VPN Example—Customer Provides Internet Service

IN THIS SECTION

In this example, the carrier customer is not required to configure MPLS and LDP on its network. However, the carrier provider must configure MPLS and LDP on its network.

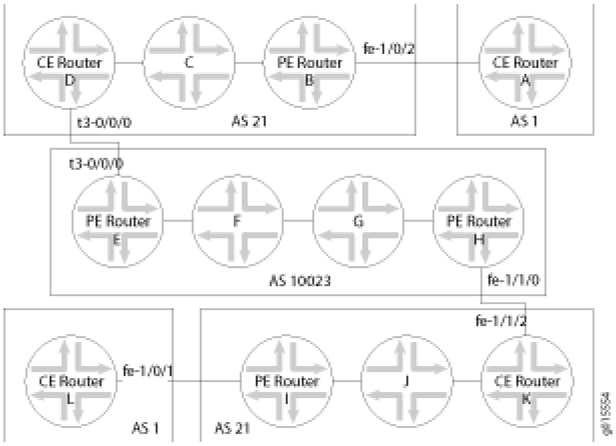
For configuration information see the following sections:

Network Topology for Carrier-of-Carriers Service

A carrier-of-carriers service allows an Internet service provider (ISP) to connect to a transparent outsourced backbone at multiple locations.

Figure 51 on page 539 shows the network topology in this carrier-of-carriers example.

Figure 51: Carrier-of-Carriers VPN Example Network Topology



Configuration for Router A

In this example, Router A represents an end customer. You configure this router as a CE device.

```
[edit]
protocols {
    bgp {
        group to-routerB {
            export attached;
            peer-as 21;
            as-override;
            neighbor 192.168.197.169;
        }
    }
}
policy-options {
    policy-statement attached {
        from protocol direct;
        then accept;
```

```
    }
}
```

Configuration for Router B

Router B can act as the gateway router, responsible for aggregating end customers and connecting them to the network. If a full-mesh IBGP session is configured, you can use route reflectors.

```
[edit]
protocols {
    bgp {
        group int {
            type internal;
            local-address 10.255.14.179;
            neighbor 10.255.14.175;
            neighbor 10.255.14.181;
            neighbor 10.255.14.176;
            neighbor 10.255.14.178;
            neighbor 10.255.14.177;
        }
        group to-vpn-blue {
            peer-as 1;
            neighbor 192.168.197.170;
        }
    }
    ospf {
        area 0.0.0.0 {
            interface lo0.0 {
                passive;
            }
            interface fe-1/0/3.0;
            interface fe-1/0/2.0 {
                passive;
            }
        }
    }
}
```

Configuration for Router C

Configure Router C:

```
[edit]
protocols {
    bgp {
        group int {
            type internal;
            local-address 10.255.14.176;
            neighbor 10.255.14.179;
            neighbor 10.255.14.175;
            neighbor 10.255.14.177;
            neighbor 10.255.14.178;
            neighbor 10.255.14.181;
        }
    }
    ospf {
        area 0.0.0.0 {
            interface lo0.0 {
                passive;
            }
            interface fe-0/3/3.0;
            interface fe-0/3/0.0;
        }
    }
}
```

Configuration for Router D

Router D is the CE router with respect to AS 10023. In a carrier-of-carriers VPN, the CE router must be able to send labels to the carrier provider; this is done with the `labeled-unicast` statement in group `to-isp-red`.

```
[edit]
protocols {
    mpls {
        interface t3-0/0/0.0;
    }
    bgp {
        group int {
```

```
            type internal;
            local-address 10.255.14.175;
            neighbor 10.255.14.179;
            neighbor 10.255.14.176;
            neighbor 10.255.14.177;
            neighbor 10.255.14.178;
            neighbor 10.255.14.181;
        }
        group to-isp-red {
            export internal;
            peer-as 10023;
            neighbor 192.168.197.13 {
                family inet {
                    labeled-unicast;
                }
            }
        }
    }
    ospf {
        area 0.0.0.0 {
            interface lo0.0 {
                passive;
            }
            interface fe-0/3/0.0;
            interface t3-0/0/0.0 {
                passive;
            }
        }
    }
}
policy options {
    policy-statement internal {
        term a {
            from protocol [ ospf direct ];
            then accept;
        }
        term b {
            then reject;
        }
    }
}
```

Configuration for Router E

This configuration sets up the `inet-vpn` IBGP session with Router H and the PE router portion of the VPN with Router D. Because Router D is required to send labels in this example, configure the BGP session with the `labeled-unicast` statement within the virtual routing and forwarding (*VRF*) table.

```
[edit]
protocols {
    mpls {
        interface t3-0/2/0.0;
        interface at-0/1/0.0;
    }
    bgp {
        group pe-pe {
            type internal;
            local-address 10.255.14.171;
            family inet-vpn {
                any;
            }
            neighbor 10.255.14.173;
        }
    }
    isis {
        interface at-0/1/0.0;
        interface lo0.0 {
            passive;
        }
    }
    ldp {
        interface at-0/1/0.0;
    }
}
routing-instances {
    vpn-isp1 {
        instance-type vrf;
        interface t3-0/2/0.0;
        route-distinguisher 10.255.14.171:21;
        vrf-import vpn-isp1-import;
        vrf-export vpn-isp1-export;
        protocols {
            bgp {
                group to-isp1 {
```

```
                    peer-as 21;
                    neighbor 192.168.197.14 {
                        family inet {
                            labeled-unicast;
                        }
                    }
                }
            }
        }
    }
}
policy-options {
    policy-statement vpn-isp1-import {
        term a {
            from {
                protocol bgp;
                community vpn-isp1-comm;
            }
            then accept;
        }
        term b {
            then reject;
        }
    }
    policy-statement vpn-isp1-export {
        term a {
            from protocol bgp;
            then {
                community add vpn-isp1-comm;
                accept;
            }
        }
        term b {
            then reject;
        }
    }
    community vpn-isp1-comm members target:69:21;
}
```

Configuration for Router F

Configure Router F to act as a label-swapping router:

```
[edit]
protocols {
    isis {
        interface so-0/2/0.0;
        interface at-0/3/0.0;
        interface lo0.0 {
            passive;
        }
    }
    ldp {
        interface so-0/2/0.0;
        interface at-0/3/0.0;
    }
}
```

Configuration for Router G

Configure Router G to act as a label-swapping router:

```
[edit]
protocols {
    isis {
        interface so-0/0/0.0;
        interface so-1/0/0.0;
        interface lo0.0 {
            passive;
        }
    }
    ldp {
        interface so-0/0/0.0;
        interface so-1/0/0.0;
    }
}
```

Configuration for Router H

Router H acts as the PE router for AS 10023. The configuration that follows is similar to that for Router F:

```
[edit]
protocols {
    mpls {
        interface fe-1/1/0.0;
        interface so-1/0/0.0;
    }
    bgp {
        group pe-pe {
            type internal;
            local-address 10.255.14.173;
            family inet-vpn {
                any;
            }
            neighbor 10.255.14.171;
        }
    }
    isis {
        interface so-1/0/0.0;
        interface lo0.0 {
            passive;
        }
    }
    ldp {
        interface so-1/0/0.0;
    }
}
routing-instances {
    vpn-isp1 {
        instance-type vrf;
        interface fe-1/1/0.0;
        route-distinguisher 10.255.14.173:21;
        vrf-import vpn-isp1-import;
        vrf-export vpn-isp1-export;
        protocols {
            bgp {
                group to-isp1 {
                    peer-as 21;
```

```
                    neighbor 192.168.197.94 {
                        family inet {
                            labeled-unicast;
                        }
                    }
                }
            }
        }
    }
}
policy-options {
    policy-statement vpn-isp1-import {
        term a {
            from {
                protocol bgp;
                community vpn-isp1-comm;
            }
            then accept;
        }
        term b {
            then reject;
        }
    }
    policy-statement vpn-isp1-export {
        term a {
            from protocol bgp;
            then {
                community add vpn-isp1-comm;
                accept;
            }
        }
        term b {
            then reject;
        }
    }
    community vpn-isp1-comm members target:69:21;
}
```

Configuration for Router I

Configure Router I to connect to the basic Internet service customer (Router L):

```
[edit]
protocols {
    mpls {
        interface fe-1/0/1.0;
        interface fe-1/1/3.0;
    }
    bgp {
        group int {
            type internal;
            local-address 10.255.14.181;
            neighbor 10.255.14.177;
            neighbor 10.255.14.179;
            neighbor 10.255.14.175;
            neighbor 10.255.14.176;
            neighbor 10.255.14.178;
        }
        group to-vpn-green {
            peer-as 1;
            neighbor 192.168.197.198;
        }
    }
    ospf {
        area 0.0.0.0 {
            interface lo0.0 {
                passive;
            }
            interface fe-1/0/1.0 {
                passive;
            }
            interface fe-1/1/3.0;
        }
    }
}
```

Configuration for Router J

Configure Router J as a label-swapping router:

```
[edit]
protocols {
    bgp {
        group int {
            type internal;
            local-address 10.255.14.178;
            neighbor 10.255.14.177;
            neighbor 10.255.14.181;
            neighbor 10.255.14.175;
            neighbor 10.255.14.176;
            neighbor 10.255.14.179;
        }
    }
}
ospf {
    area 0.0.0.0 {
        interface lo0.0 {
            passive;
        }
        interface fe-1/0/2.0;
        interface fe-1/0/3.0;
    }
}
```

Configuration for Router K

Router K acts as the CE router at the end of the connection to the carrier provider. As in the configuration for Router D, include the `labeled-unicast` statement for the EBGP session:

```
[edit]
protocols {
    mpls {
        interface fe-1/1/2.0;
        interface fe-1/0/2.0;
    }
    bgp {
        group int {
```

```
            type internal;
            local-address 10.255.14.177;
            neighbor 10.255.14.181;
            neighbor 10.255.14.178;
            neighbor 10.255.14.175;
            neighbor 10.255.14.176;
            neighbor 10.255.14.179;
        }
        group to-isp-red {
            export internal;
            peer-as 10023;
            neighbor 192.168.197.93 {
                family inet {
                    labeled-unicast;
                }
            }
        }
    }
    ospf {
        area 0.0.0.0 {
            interface lo0.0 {
                passive;
            }
            interface fe-1/0/2.0;
            interface fe-1/1/2.0 {
                passive;
            }
        }
    }
}
policy-options {
    policy-statement internal {
        term a {
            from protocol [ ospf direct ];
            then accept;
        }
        term b {
            then reject;
        }
    }
}
```

Configuration for Router L

Configure Router L to act as the end customer for the carrier-of-carriers VPN service:

```
[edit]
protocols {
    bgp {
        group to-routerI {
            export attached;
            peer-as 21;
            neighbor 192.168.197.197;
        }
    }
}
policy-options {
    policy-statement attached {
        from protocol direct;
        then accept;
    }
}
```

SEE ALSO

MPLS Feature Support on QFX Series and EX4600 Switches

Configuring Carrier-of-Carriers VPNs for Customers That Provide VPN Service

IN THIS SECTION

You can configure a carrier-of-carriers VPN service for customers who want VPN service.

To configure the routers (or switches) in the customer’s and provider’s networks to enable carrier-of-carriers VPN service, perform the steps in the following sections:

Configuring the Carrier-of-Carriers Customer's PE Router

The carrier-of-carriers customer's PE router (or switch) is connected to the end customer's CE router (or switch).

The following sections describe how to configure the carrier-of-carriers customer's PE router (or switch):

Configuring MPLS

To configure MPLS on the carrier-of-carriers customer's PE router (or switch), include the `mpls` statement:

```
mpls {
    interface interface-name;
    interface interface-name;
}
```

You can include this statement at the following hierarchy levels:

- `[edit protocols]`
- `[edit logical-systems logical-system-name protocols]`

Configuring BGP

Include the `labeled-unicast` statement in the configuration for the IBGP session to the carrier-of-carriers customer's CE router (or switch)), and include the `family-inet-vpn` statement in the configuration for the IBGP session to the carrier-of-carriers PE router (or switch) on the other side of the network:

```
bgp {
    group group-name {
        type internal;
        local-address address;
        neighbor address {
            family inet {
                labeled-unicast;
                resolve-vpn;
            }
        }
    }
    neighbor address {
        family inet-vpn {
            any;
```

```
        }
    }
}
```

You can include these statements at the following hierarchy levels:

- `[edit protocols]`
- `[edit logical-systems logical-system-name protocols]`

Configuring OSPF

To configure OSPF on the carrier-of-carriers customer's PE router (or switch), include the `ospf` statement:

```
ospf {
    area area-id {
        interface interface-name {
            passive;
        }
        interface interface-name;
    }
}
```

You can include this statement at the following hierarchy levels:

- `[edit protocols]`
- `[edit logical-systems logical-system-name protocols]`

Configuring LDP

To configure LDP on the carrier-of-carriers customer's PE router (or switch), include the `ldp` statement:

```
ldp {
    interface interface-name;
}
```

You can include this statement at the following hierarchy levels:

- `[edit protocols]`
- `[edit logical-systems logical-system-name protocols]`

Configuring VPN Service in the Routing Instance

To configure VPN service for the end customer's CE router (or switch) on the carrier-of-carriers customer's PE router (or switch), include the following statements:

```
instance-type vrf;
interface interface-name;
route-distinguisher address;
vrf-import policy-name;
vrf-export policy-name;
protocols {
    bgp {
        group group-name {
            peer-as as-number;
            neighbor address;
        }
    }
}
```

You can include these statements at the following hierarchy levels:

- `[edit routing-instances` *`routing-instance-name`*`]`
- `[edit logical-systems` *`logical-system-name`* `routing-instances` *`routing-instance-name`*`]`

Configuring Policy Options

To configure policy options to import and export routes to and from the end customer's CE router (or switch), include the `policy-statement` and `community` statements:

```
policy-statement policy-name {
    term term-name {
        from {
            protocol bgp;
            community community-name;
        }
        then accept;
    }
    term term-name {
        then reject;
    }
}
```

```
policy-statement policy-name {
    term term-name {
        from protocol bgp;
        then {
            community add community-name;
            accept;
        }
    }
    term term-name {
        then reject;
    }
}
community community-name members value;
```

You can include these statements at the following hierarchy levels:

- `[edit policy-options]`
- `[edit logical-systems logical-system-name policy-options]`

Configuring the Carrier-of-Carriers Customer's CE Router (or switch)

The carrier-of-carriers customer's CE router (or switch) connects to the provider's PE router (or switch). Complete the instructions in the following sections to configure the carrier-of-carriers customers' CE router (or switch):

Configuring MPLS

In the MPLS configuration for the carrier-of-carriers customer's CE router (or switch), include the interfaces to the provider's PE router (or switch) and to a P router (or switch) in the customer's network:

```
mpls {
    traffic-engineering bgp-igp;
    interface interface-name;
    interface interface-name;
}
```

You can include these statements at the following hierarchy levels:

- `[edit protocols]`
- `[edit logical-systems logical-system-name protocols]`

Configuring BGP

In the BGP configuration for the carrier-of-carriers customer’s CE router (or switch), configure a group that includes the `labeled-unicast` statement to extend VPN service to the PE router (or switch)connected to the end customer’s CE router (or switch):

```
bgp {
    group group-name {
        type internal;
        local-address address;
        neighbor address {
            family inet {
                labeled-unicast;
            }
        }
    }
}
```

You can include the `bgp` statement at the following hierarchy levels:

- `[edit protocols]`
- `[edit logical-systems logical-system-name protocols]`

To configure a group to send labeled internal routes to the provider’s PE router (or switch), include the `bgp` statement:

```
bgp {
    group group-name {
        export internal;
        peer-as as-number;
        neighbor address {
            family inet {
                labeled-unicast;
            }
        }
    }
}
```

You can include this statement at the following hierarchy levels:

- `[edit protocols]`

- `[edit logical-systems logical-system-name protocols]`

Configuring OSPF and LDP

To configure OSPF and LDP on the carrier-of-carriers customer's CE router (or switch), include the `ospf` and `ldp` statements:

```
ospf {
    area area-id {
        interface interface-name {
            passive;
        }
        interface interface-name;
    }
}
ldp {
    interface interface-name;
}
```

You can include these statements at the following hierarchy levels:

- `[edit protocols]`
- `[edit logical-systems logical-system-name protocols]`

Configuring Policy Options

To configure the policy options on the carrier-of-carriers customer's CE router (or switch), include the `policy-statement` statement:

```
policy-statement policy-statement-name {
    term term-name {
        from protocol [ ospf direct ldp ];
        then accept;
    }
    term term-name {
        then reject;
    }
}
```

You can include this statement at the following hierarchy levels:

- `[edit policy-options]`
- `[edit logical-systems *logical-system-name* policy-options]`

Configuring the Provider's PE Router or Switch

The carrier-of-carriers provider's PE routers (or switches) connect to the carrier customer's CE routers (or switches) . Complete the instructions in the following sections to configure the provider's PE router (or switch):

Configuring MPLS

In the MPLS configuration, specify at least two interfaces—one to the customer's CE router (or switch)and one to connect to the provider's PE router (or switch)on the other side of the provider's network:

```
interface interface-name;
interface interface-name;
```

You can include these statements at the following hierarchy levels:

- `[edit protocols mpls]`
- `[edit logical-systems *logical-system-name* protocols mpls]`

Configuring a PE-to-PE BGP Session

To configure a PE-to-PE BGP session on the provider's PE routers (or switches) to allow VPN-IPv4 routes to pass between the PE routers (or switches, include the `bgp` statement:

```
bgp {
    group group-name {
        type internal;
        local-address address;
        family inet-vpn {
            any;
        }
        neighbor address;
    }
}
```

You can include this statement at the following hierarchy levels:

- `[edit protocols]`
- `[edit logical-systems logical-system-name protocols]`

Configuring IS-IS and LDP

To configure IS-IS and LDP on the provider's PE routers (or switches), include the `isis` and `ldp` statements:

```
isis {
    interface interface-name;
    interface interface-name {
        passive;
    }
}
ldp {
    interface interface-name;
}
```

You can include these statements at the following hierarchy levels:

- `[edit protocols]`
- `[edit logical-systems logical-system-name protocols]`

Configuring Policy Options

To configure policy statements on the provider's PE router (or switch) to export routes to and import routes from the carrier customer's network, include the `policy-statement` and `community` statements:

```
policy-statement statement-name {
    term term-name {
        from {
            protocol bgp;
            community community-name;
        }
        then accept;
    }
    term term-name {
        then reject;
    }
}
```

```
policy-statement statement-name {
    term term-name {
        from protocol bgp;
        then {
            community add community-name;
            accept;
        }
    }
    term term-name {
        then reject;
    }
}
community community-name members value;
```

You can include these statements at the following hierarchy levels:

- `[edit policy-options]`
- `[edit logical-systems logical-system-name policy-options]`

Configuring a Routing Instance to Send Routes to the CE Router

To configure the routing instance on the provider's PE router (or switch) to send labeled routes to the carrier customer's CE router (or switch), include the following statements:

```
instance-type vrf;
interface interface-name;
route-distinguisher value;
vrf-import policy-name;
vrf-export policy-name;
protocols {
    bgp {
        group group-name {
            peer-as as-number;
            neighbor address {
                family inet {
                    labeled-unicast;
                }
            }
        }
    }
}
```

You can include these statements at the following hierarchy levels:

- `[edit routing-instances routing-instance-name]`
- `[edit logical-systems logical-system-name routing-instances routing-instance-name]`

SEE ALSO

MPLS Feature Support on QFX Series and EX4600 Switches

Understanding Interprovider and Carrier-of-Carriers VPNs | 444

Carrier-of-Carriers VPN Example—Customer Provides VPN Service

IN THIS SECTION

In this example, the carrier customer *must* run some form of MPLS (Resource Reservation Protocol [RSVP] or LDP) on its network to provide VPN services to the end customer. In the example below, Router B and Router I act as PE routers (or switches), and a functioning MPLS path is required between these routers if they exchange VPN-IPv4 routes.

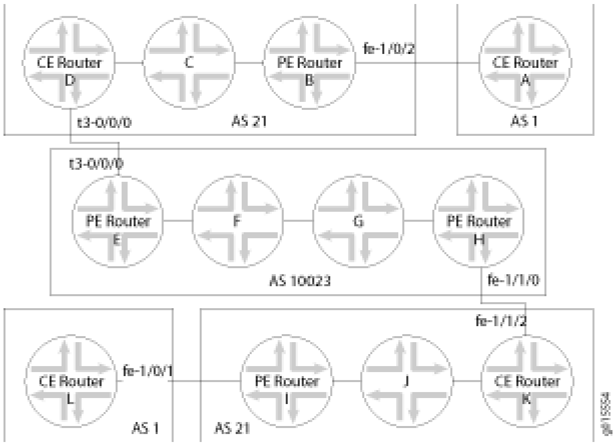
For configuration information see the following sections:

Network Topology for Carrier-of-Carriers Service

A carrier-of-carriers service allows an Internet service provider (ISP) to connect to a transparent outsourced backbone at multiple locations.

Figure 52 on page 562 shows the network topology in this carrier-of-carriers example.

Figure 52: Carrier-of-Carriers VPN Example Network Topology



Configuration for Router A

In this example, Router A acts as the CE router for the end customer. Configure a default `family inet` BGP session on Router A:

```
[edit]
protocols {
    bgp {
        group to-routerB {
            export attached;
            peer-as 21;
            neighbor 192.168.197.169;
        }
    }
}
policy-options {
    policy-statement attached {
        from protocol direct;
        then accept;
```

```
    }
}
```

Configuration for Router B

Because Router B is the PE router for the end customer CE router (Router A), you need to configure a routing instance (`vpna`). Configure the `labeled-unicast` statement on the IBGP session to Router D, and configure `family-inet-vpn` for the IBGP session to the other side of the network with Router I:

```
[edit]
protocols {
    mpls {
        interface fe-1/0/2.0;
        interface fe-1/0/3.0;
    }
    bgp {
        group int {
            type internal;
            local-address 10.255.14.179;
            neighbor 10.255.14.175 {
                family inet {
                    labeled-unicast {
                        resolve-vpn;
                    }
                }
            }
        }
        neighbor 10.255.14.181 {
            family inet-vpn {
                any;
            }
        }
    }
    ospf {
        area 0.0.0.0 {
            interface lo0.0 {
                passive;
            }
            interface fe-1/0/3.0;
        }
    }
```

```
    ldp {
        interface fe-1/0/3.0;
    }
}
routing-instances {
    vpna {
        instance-type vrf;
        interface fe-1/0/2.0;
        route-distinguisher 10.255.14.179:21;
        vrf-import vpna-import;
        vrf-export vpna-export;
        protocols {
            bgp {
                group vpna-06 {
                    peer-as 1;
                    neighbor 192.168.197.170;
                }
            }
        }
    }
}
policy-options {
    policy-statement vpna-import {
        term a {
            from {
                protocol bgp;
                community vpna-comm;
            }
            then accept;
        }
        term b {
            then reject;
        }
    }
    policy-statement vpna-export {
        term a {
            from protocol bgp;
            then {
                community add vpna-comm;
                accept;
            }
        }
        term b {
```

```
            then reject;
        }
    }
    community vpna-comm members target:100:1001;
}
```

Configuration for Router C

Configure Router C as a label-swapping router within the local AS:

```
[edit]
protocols {
    mpls {
        traffic-engineering bgp-igp;
    }
    ospf {
        area 0.0.0.0 {
            interface lo0.0 {
                passive;
            }
            interface fe-0/3/3.0;
            interface fe-0/3/0.0;
        }
    }
    ldp {
        interface fe-0/3/0.0;
        interface fe-0/3/3.0;
    }
}
```

Configuration for Router D

Router D acts as the CE router for the VPN services provided by the AS 10023 network. In the BGP group configuration for group `int`, which handles traffic to Router B (10.255.14.179), you include the `labeled-unicast` statement. You also need to configure the BGP group `to-isp-red` to send labeled internal routes to the PE router (Router E).

```
[edit]
protocols {
    mpls {
```

```
        traffic-engineering bgp-igp;
        interface fe-0/3/0.0;
        interface t3-0/0/0.0;
    }
    bgp {
        group int {
            type internal;
            local-address 10.255.14.175;
            neighbor 10.255.14.179 {
                family inet {
                    labeled-unicast;
                }
            }
        }
        group to-isp-red {
            export internal;
            peer-as 10023;
            neighbor 192.168.197.13 {
                family inet {
                    labeled-unicast;
                }
            }
        }
    }
    ospf {
        area 0.0.0.0 {
            interface lo0.0 {
                passive;
            }
            interface fe-0/3/0.0;
        }
    }
    ldp {
        interface fe-0/3/0.0;
    }
}
policy-options {
    policy-statement internal {
        term a {
            from protocol [ ospf direct ];
            then accept;
        }
        term b {
```

```
            then reject;
        }
    }
}
```

Configuration for Router E

Router E and Router H are PE routers. Configure a PE-router-to-PE-router BGP session to allow VPN-IPv4 routes to pass between these two PE routers. Configure the routing instance on Router E to send labeled routes to the CE router (Router D).

Configure Router E:

```
[edit]
protocols {
    mpls {
        interface t3-0/2/0.0;
        interface at-0/1/0.0;
    }
    bgp {
        group pe-pe {
            type internal;
            local-address 10.255.14.171;
            family inet-vpn {
                any;
            }
            neighbor 10.255.14.173;
        }
    }
    isis {
        interface at-0/1/0.0;
        interface lo0.0 {
            passive;
        }
    }
    ldp {
        interface at-0/1/0.0;
    }
}
policy-options {
    policy-statement vpn-isp1-import {
        term a {
```

```
            from {
                protocol bgp;
                community vpn-isp1-comm;
            }
            then accept;
        }
        term b {
            then reject;
        }
    }
    policy-statement vpn-isp1-export {
        term a {
            from protocol bgp;
            then {
                community add vpn-isp1-comm;
                accept;
            }
        }
        term b {
            then reject;
        }
    }
    community vpn-isp1-comm members target:69:21;
}
routing-instances {
    vpn-isp1 {
        instance-type vrf;
        interface t3-0/2/0.0;
        route-distinguisher 10.255.14.171:21;
        vrf-import vpn-isp1-import;
        vrf-export vpn-isp1-export;
        protocols {
            bgp {
                group to-isp1 {
                    peer-as 21;
                    neighbor 192.168.197.14 {
                        as-override;
                        family inet {
                            labeled-unicast;
                        }
                    }
                }
            }
```

```
        }
    }
}
```

Configuration for Router F

Configure Router F to swap labels for routes running through its interfaces:

```
[edit]
protocols {
    isis {
        interface so-0/2/0.0;
        interface at-0/3/0.0;
        interface lo0.0 {
            passive;
        }
    }
    ldp {
        interface so-0/2/0.0;
        interface at-0/3/0.0;
    }
}
```

Configuration for Router G

Configure Router G:

```
[edit]
protocols {
    isis {
        interface so-0/0/0.0;
        interface so-1/0/0.0;
        interface lo0.0 {
            passive;
        }
    }
    ldp {
        interface so-0/0/0.0;
        interface so-1/0/0.0;
```

```
    }
}
```

Configuration for Router H

The configuration for Router H is similar to the configuration for Router E:

```
[edit]
protocols {
    mpls {
        interface fe-1/1/0.0;
        interface so-1/0/0.0;
    }
    bgp {
        group pe-pe {
            type internal;
            local-address 10.255.14.173;
            family inet-vpn {
                any;
            }
            neighbor 10.255.14.171;
        }
    }
    isis {
        interface so-1/0/0.0;
        interface lo0.0 {
            passive;
        }
    }
    ldp {
        interface so-1/0/0.0;
    }
}
routing-instances {
    vpn-isp1 {
        instance-type vrf;
        interface fe-1/1/0.0;
        route-distinguisher 10.255.14.173:21;
        vrf-import vpn-isp1-import;
        vrf-export vpn-isp1-export;
        protocols {
```

```
            bgp {
                group to-isp1 {
                    peer-as 21;
                    neighbor 192.168.197.94 {
                        as-override;
                        family inet {
                            labeled-unicast;
                        }
                    }
                }
            }
        }
    }
}
policy-options {
    policy-statement vpn-isp1-import {
        term a {
            from {
                protocol bgp;
                community vpn-isp1-comm;
            }
            then accept;
        }
        term b {
            then reject;
        }
    }
    policy-statement vpn-isp1-export {
        term a {
            from protocol bgp;
            then {
                community add vpn-isp1-comm;
                accept;
            }
        }
        term b {
            then reject;
        }
    }
    community vpn-isp1-comm members target:69:21;
}
```

Configuration for Router I

Router I acts as the PE router for the end customer. The configuration that follows is similar to the configuration for Router B:

```
[edit]
protocols {
    mpls {
        interface fe-1/0/1.0;
        interface fe-1/1/3.0;
    }
    bgp {
        group int {
            type internal;
            local-address 10.255.14.181;
            neighbor 10.255.14.177 {
                family inet {
                    labeled-unicast {
                        resolve-vpn;
                    }
                }
            }
            neighbor 10.255.14.179 {
                family inet-vpn {
                    any;
                }
            }
        }
    }
    ospf {
        area 0.0.0.0 {
            interface lo0.0 {
                passive;
            }
            interface fe-1/1/3.0;
        }
    }
    ldp {
        interface fe-1/1/3.0;
    }
}
routing-instances {
```

```
    vpna {
        instance-type vrf;
        interface fe-1/0/1.0;
        route-distinguisher 10.255.14.181:21;
        vrf-import vpna-import;
        vrf-export vpna-export;
        protocols {
            bgp {
                group vpna-0 {
                    peer-as 1;
                    neighbor 192.168.197.198;
                }
            }
        }
    }
}
policy-options {
    policy-statement vpna-import {
        term a {
            from {
                protocol bgp;
                community vpna-comm;
            }
            then accept;
        }
        term b {
            then reject;
        }
    }
    policy-statement vpna-export {
        term a {
            from protocol bgp;
            then {
                community add vpna-comm;
                accept;
            }
        }
        term b {
            then reject;
        }
    }
```

```
    community vpna-comm members target:100:1001;
}
```

Configuration for Router J

Configure Router J to swap labels for routes running through its interfaces:

```
[edit]
protocols {
    mpls {
        traffic-engineering bgp-igp;
    }
    ospf {
        area 0.0.0.0 {
            interface lo0.0 {
                passive;
            }
            interface fe-1/0/2.0;
            interface fe-1/0/3.0;
        }
    }
    ldp {
        interface fe-1/0/2.0;
        interface fe-1/0/3.0;
    }
}
```

Configuration for Router K

The configuration for Router K is similar to the configuration for Router D:

```
[edit]
protocols {
    mpls {
        traffic-engineering bgp-igp;
        interface fe-1/1/2.0;
        interface fe-1/0/2.0;
    }
    bgp {
        group int {
```

```
            type internal;
            local-address 10.255.14.177;
            neighbor 10.255.14.181 {
                family inet {
                    labeled-unicast;
                }
            }
        }
        group to-isp-red {
            export internal;
            peer-as 10023;
            neighbor 192.168.197.93 {
                family inet {
                    labeled-unicast;
                }
            }
        }
    }
    ospf {
        area 0.0.0.0 {
            interface lo0.0 {
                passive;
            }
            interface fe-1/0/2.0;
        }
    }
    ldp {
        interface fe-1/0/2.0;
    }
}
policy-options {
    policy-statement internal {
        term a {
            from protocol [ ospf direct ];
            then accept;
        }
        term b {
            then reject;
        }
    }
}
```

Configuration for Router L

In this example, Router L is the end customer’s CE router. Configure a default family `inet` BGP session on Router L:

```
[edit]
protocols {
    bgp {
        group to-I {
            export attached;
            peer-as 21;
            neighbor 192.168.197.197;
        }
    }
}
policy-options {
    policy-statement attached {
        from protocol direct;
        then accept;
    }
}
```

SEE ALSO

MPLS Feature Support on QFX Series and EX4600 Switches

Multiple Instances for LDP and Carrier-of-Carriers VPNs

By configuring multiple LDP routing instances, you can use LDP to advertise labels in a carrier-of-carriers VPN from a core provider PE router to a customer carrier CE router. Having LDP advertise labels in this manner is especially useful when the carrier customer is a basic ISP and wants to restrict full Internet routes to its PE routers. By using LDP instead of BGP, the carrier customer shields its other internal routers from the Internet at large. Multiple-instance LDP is also useful when a carrier customer wants to provide Layer 3 VPN or Layer 2 VPN services to its customers.

For an example of how to configure multiple LDP routing instances for carrier-of-carriers VPNs see https://www.juniper.net/documentation/us/en/software/junos/mpls/topics/example/multiple-instance-ldp-configuring-detailed-solutions.html.

SEE ALSO

MPLS Feature Support on QFX Series and EX4600 Switches

Change History Table

Feature support is determined by the platform and release you are using. Use Feature Explorer to determine if a feature is supported on your platform.

Release	Description
17.1R1	Support for VPN service as the customer is supported on QFX10000 switches starting with Junos OS Release 17.1R1.

CHAPTER 4

Multicast on Layer 3 VPNs

IN THIS CHAPTER

Multicast on Layer 3 VPNs

IN THIS SECTION

You can configure multicast routing over a network running a Layer 3 VPN that complies with RFC 4364. This topic provides an overview of multicast and describes configuring devices to support multicast traffic in a Layer 3 VPN.

Understanding MVPN Concepts and Protocols

IN THIS SECTION

Multicast over Layer 3 VPNs Overview

In the unicast environment for Layer 3 VPNs, all VPN state information is contained within the PE routers. However, with multicast for Layer 3 VPNs, Protocol Independent Multicast (PIM) adjacencies are established in one of the following ways:

- You can set PIM adjacencies between the CE router and the PE router through a VRF instance at the `[edit routing-instances` *`instance-name`* `protocols pim]` hierarchy level. You must include the `group-address` statement for the provider tunnel, specifying a multicast group. The rendezvous point (RP) listed within the VRF-instance is the VPN customer RP (C-RP).

- You can also set the primary PIM instance and the PE's IGP neighbors by configuring statements at the [edit protocols pim] hierarchy level. You must add the multicast group specified in the VRF instance to the primary PIM instance. The set of primary PIM adjacencies throughout the service provider network makes up the forwarding path that becomes an RP tree rooted at the service provider RP (SP-RP). Therefore, P routers within the provider core must maintain multicast state information for the VPNs.

For this to work properly, you need two types of RP routers for each VPN:

- A C-RP—An RP router located somewhere within the VPN (can be either a service provider router or a customer router).

- An SP-RP—An RP router located within the service provider network.

NOTE: A PE router can act as the SP-RP and the C-RP. Moving these multicast configuration tasks to service provider routers helps to simplify the multicast Layer 3 VPN configuration process for customers. However, configuration of both SP-RP and VPN C-RP on the same PE router is not supported.

To configure multicast over a Layer 3 VPN, you must install a Tunnel Services *Physical Interface Card* (PIC) on the following devices:

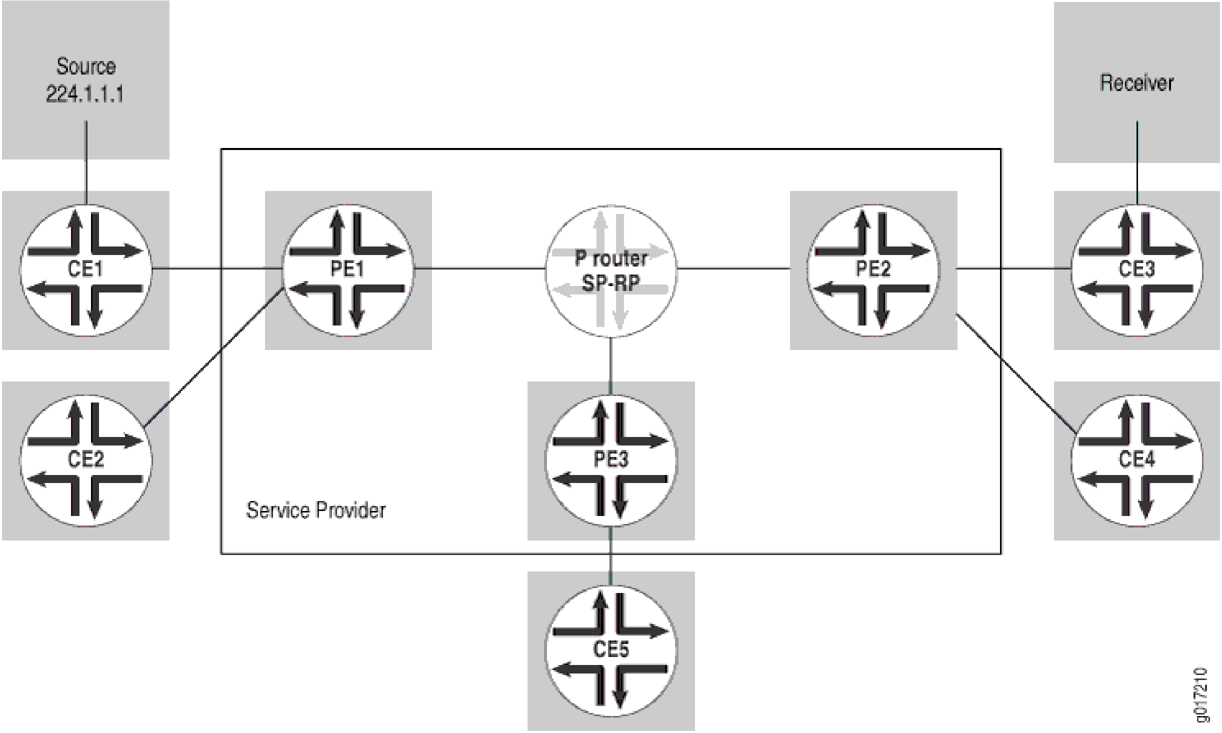
- P routers acting as RPs
- PE routers configured to run multicast routing
- CE routers acting as designated routers or as VPN-RPs

For more information about running multicast over Layer 3 VPNs, see the following documents:

- Internet draft draft-rosen-vpn-mcast-02.txt, *Multicast in MPLS/BGP VPNs*
- Multicast Protocols User Guide

The sections that follow describe the operation of a multicast VPN. Figure 53 on page 581 illustrates the network topology used.

Figure 53: Multicast Topology Overview



Sending PIM Hello Messages to the PE Routers

The first step in initializing multicast over a Layer 3 VPN is the distribution of a PIM Hello message from a PE router (called PE3 in this section) to all the other PE routers on which PIM is configured.

You configure PIM on the Layer 3 VPN routing instance on the PE3 router. If a Tunnel Services PIC is installed in the routing platform, a multicast interface is created. This interface is used to communicate between the PIM instance within the VRF routing instance and the primary PIM instance.

The following occurs when a PIM Hello message is sent to the PE routers:

1. A PIM Hello message is sent from the VRF routing instance over the multicast interface. A generic routing encapsulation (GRE) header is prepended to the PIM Hello message. The header message includes the VPN group address and the loopback address of the PE3 router.

2. A PIM register header is prepended to the Hello message as the packet is looped through the PIM encapsulation interface. This header contains the destination address of the SP-RP and the loopback address of the PE3 router.

3. The packet is sent to the SP-RP.

4. The SP-RP removes the top header from the packet and sends the remaining GRE-encapsulated Hello message to all the PE routers.

5. The primary PIM instance on each PE router handles the GRE encapsulated packet. Because the VPN group address is contained in the packet, the primary instance removes the GRE header from the packet and sends the Hello message, which contains the proper VPN group address within the VRF routing instance, over the multicast interface.

Sending PIM Join Messages to the PE Routers

To receive a multicast broadcast from a multicast network, a CE router must send a PIM Join message to the C-RP. The process described in this section refers to Figure 53 on page 581.

The CE5 router needs to receive a multicast broadcast from multicast source 224.1.1.1. To receive the broadcast, it sends a PIM Join message to the C-RP (the PE3 router):

1. The PIM Join message is sent through the multicast interface, and a GRE header is prepended to the message. The GRE header contains the VPN group ID and the loopback address of the PE3 router.
2. The PIM Join message is then sent through the PIM encapsulation interface and a register header is prepended to the packet. The register header contains the IP address of the SP-RP and the loopback address of the PE3 router.
3. The PIM Join message is sent to the SP-RP by means of unicast routing.
4. On the SP-RP, the register header is stripped off (the GRE header remains) and the packet is sent to all the PE routers.
5. The PE2 router receives the packet, and because the link to the C-RP is through the PE2 router, it sends the packet through the multicast interface to remove the GRE header.
6. Finally, the PIM Join message is sent to the C-RP.

Receiving the Multicast Transmission

The steps that follow outline how a multicast transmission is propagated across the network:

1. The multicast source connected to the CE1 router sends the packet to group 224.1.1.1 (the VPN group address). The packet is encapsulated into a PIM register.
2. Because this packet already includes the PIM header, it is forwarded by means of unicast routing to the C-RP over the Layer 3 VPN.
3. The C-RP removes the packet and sends it out the downstream interfaces (which include the interface back to the CE3 router). The CE3 router also forwards this to the PE3 router.
4. The packet is sent through the multicast interface on the PE2 router; in the process, the GRE header is prepended to the packet.

5. Next, the packet is sent through the PIM encapsulation interface, where the register header is prepended to the data packet.

6. The packet is then forwarded to the SP-RP, which removes the register header, leaves the GRE header intact, and sends the packet to the PE routers.

7. PE routers remove the GRE header and forward the packet to the CE routers that requested the multicast broadcast by sending the PIM Join message.

> **NOTE**: PE routers that have not received requests for multicast broadcasts from their connected CE routers still receive packets for the broadcast. These PE routers drop the packets as they are received.

Supported Multicast VPN Standards

Junos OS substantially supports the following RFCs and Internet draft, which define standards for multicast virtual private networks (VPNs).

- RFC 6513, *Multicast in MPLS/BGP IP VPNs*
- RFC 6514, *BGP Encodings and Procedures for Multicast in MPLS/BGP IP VPNs*
- RFC 6515, *IPv4 and IPv6 Infrastructure Addresses in BGP Updates for Multicast VPN*
- RFC 6625, *Wildcards in Multicast VPN Auto-Discovery Routes*
- Internet draft draft-morin-l3vpn-mvpn-fast-failover-06.txt, *Multicast VPN Fast Upstream Failover*
- Internet draft draft-raggarwa-l3vpn-bgp-mvpn-extranet-08.txt, *Extranet in BGP Multicast VPN (MVPN)*
- RFC 7900, *Extranet Multicast in BGP/IP MPLS VPNs (partial support)*
- RFC 8534, *Explicit Tracking with Wildcard Routes in Multicast VPN (partial support)*
- RFC 9081, *Interoperation between Multicast Virtual Private Network (MVPN) and Multicast Source Directory Protocol (MSDP) Source-Active Routes*

SEE ALSO

Supported Carrier-of-Carriers and Interprovider VPN Standards | 446

Supported VPWS Standards

Supported Layer 2 VPN Standards

Supported Layer 3 VPN Standards | 10

Configuring Multicast Layer 3 VPNs

You can configure two types of multicast Layer 3 VPNs using the Junos OS:

- Draft Rosen multicast VPNs—Draft Rosen multicast VPNs are described in RFC 4364, *BGP/MPLS IP Virtual Private Networks (VPNs)* and based on Section Two of the IETF Internet draft draft-rosen-vpn-mcast-06.txt, *Multicast in MPLS/BGP VPNs* (expired April 2004).

- Next generation multicast VPNs—Next generation multicast VPNs are described in Internet drafts draft-ietf-l3vpn-2547bis-mcast-bgp-03.txt, *BGP Encodings for Multicast in MPLS/BGP IP VPNs* and draft-ietf-l3vpn-2547bis-mcast-02.txt, *Multicast in MPLS/BGP IP VPNs*.

This section describes how to configure draft Rosen multicast VPNs. This information is provided to you in case you already have dual PIM multicast VPNs configured on your network. For information about BGP MPLS multicast VPNs (also known as next generation multicast VPNs), see "MBGP Multicast VPN Sites" on page 597.

NOTE: Draft-rosen multicast VPNs are not supported in a logical system environment even though the configuration statements can be configured under the logical-systems hierarchy.

You can configure a Layer 3 VPN to support multicast traffic using the Protocol Independent Multicast (PIM) routing protocol. To support multicast, you need to configure PIM on routers within the VPN and within the service provider’s network.

Each PE router configured to run multicast over Layer 3 VPNs must have a Tunnel Services PIC. A Tunnel Services PIC is also required on the P routers that act as rendezvous points (RPs). Tunnel Services PICs are also needed on all the CE routers acting as designated routers (first-hop/last-hop routers) or as RPs, just as they are in non-VPN PIM environments.

Configure the master PIM instance at the `[edit protocols pim]` hierarchy level on the CE and PE routers. This master PIM instance configuration on the PE router should match the configuration on the service providers core routers.

You also need to configure a PIM instance for the Layer 3 VPN at the `[edit routing-instances` *`routing-instance-name`* `protocols pim]` hierarchy level on the PE router. This creates a PIM instance for the indicated routing instance. The configuration of the PIM instance on the PE router should match the PIM instance configured on the CE router the PE router is connected to.

For information about how to configure PIM, see the Multicast Protocols User Guide .

Include the `vpn-apply-export` statement to configure the group address designated for the VPN in the service provider's network. This address must be unique for each VPN and configured on the VRF routing instance of all PE routers connecting to the same VPN. It ensures that multicast traffic is transmitted only to the specified VPN.

Include the `vpn-apply-export` statement:

```
vpn-apply-export address;
```

For a list of hierarchy levels at which you can configure this statement, see the statement summary section for this statement.

You can include this statement at the following hierarchy levels:

- `[edit routing-instances routing-instance-name protocols pim]`
- `[edit logical-systems logical-system-name routing-instances routing-instance-name protocols pim]`

The rest of the Layer 3 VPN configuration for multicast is conventional and is described in other sections of this manual. Most of the specific configuration tasks needed to activate multicast in a VPN environment involve PIM.

SEE ALSO

> Multicast Protocols User Guide

Example: Configuring PIM Join Load Balancing on Draft-Rosen Multicast VPN

IN THIS SECTION

This example shows how to configure multipath routing for external and internal virtual private network (VPN) routes with unequal interior gateway protocol (IGP) metrics, and Protocol Independent Multicast

(PIM) join load balancing on provider edge (PE) routers running Draft-Rosen multicast VPN (MVPN). This feature allows customer PIM (C-PIM) join messages to be load-balanced across external and internal BGP (EIBGP) upstream paths when the PE router has both external BGP (EBGP) and internal BGP (IBGP) paths toward the source or rendezvous point (RP).

Requirements

This example requires the following hardware and software components:

- Three routers that can be a combination of M Series Multiservice Edge Routers, MX Series 5G Universal Routing Platforms, or T Series Core Routers.
- Junos OS Release 12.1 or later running on all the devices.

Before you begin:

1. Configure the device interfaces.
2. Configure the following routing protocols on all PE routers:
   - OSPF
   - MPLS
   - LDP
   - PIM
   - BGP
3. Configure a multicast VPN.

Overview and Topology

Junos OS Release 12.1 and later support multipath configuration along with PIM join load balancing. This allows C-PIM join messages to be load-balanced across unequal EIBGP routes, if a PE router has EBGP and IBGP paths toward the source (or RP). In previous releases, only the active EBGP path was used to send the join messages. This feature is applicable to IPv4 C-PIM join messages.

During load balancing, if a PE router loses one or more EBGP paths toward the source (or RP), the C-PIM join messages that were previously using the EBGP path are moved to a multicast tunnel interface, and the reverse path forwarding (RPF) neighbor on the multicast tunnel interface is selected based on a hash mechanism.

On discovering the first EBGP path toward the source (or RP), only the new join messages get load-balanced across EIBGP paths, whereas the existing join messages on the multicast tunnel interface remain unaffected.

Though the primary goal for multipath PIM join load balancing is to utilize unequal EIBGP paths for multicast traffic, potential join loops can be avoided if a PE router chooses only the EBGP path when there are one or more join messages for different groups from a remote PE router. If the remote PE router's join message arrives after the PE router has already chosen IBGP as the upstream path, then the potential loops can be broken by changing the selected upstream path to EBGP.

> **NOTE**: During a graceful Routing Engine switchover (GRES), the EIBGP path selection for C-PIM join messages can vary, because the upstream interface selection is performed again for the new Routing Engine based on the join messages it receives from the CE and PE neighbors. This can lead to disruption of multicast traffic depending on the number of join messages received and the load on the network at the time of the graceful restart. However, the nonstop active routing feature is not supported and has no impact on the multicast traffic in a Draft-Rosen MVPN scenario.

In this example, PE1 and PE2 are the upstream PE routers for which the multipath PIM join load-balancing feature is configured. Routers PE1 and PE2 have one EBGP path and one IBGP path each toward the source. The Source and Receiver attached to customer edge (CE) routers are Free BSD hosts.

On PE routers that have EIBGP paths toward the source (or RP), such as PE1 and PE2, PIM join load balancing is performed as follows:

1. The existing join-count-based load balancing is performed such that the algorithm first selects the least loaded C-PIM interface. If there is equal or no load on all the C-PIM interfaces, the join messages get distributed equally across the available upstream interfaces.

   In Figure 54 on page 590, if the PE1 router receives PIM join messages from the CE2 router, and if there is equal or no load on both the EBGP and IBGP paths toward the source, the join messages get load-balanced on the EIBGP paths.

2. If the selected least loaded interface is a multicast tunnel interface, then there can be a potential join loop if the downstream list of the customer join (C-join) message already contains the multicast tunnel interface. In such a case, the least loaded interface among EBGP paths is selected as the upstream interface for the C-join message.

   Assuming that the IBGP path is the least loaded, the PE1 router sends the join messages to PE2 using the IBGP path. If PIM join messages from the PE3 router arrive on PE1, then the downstream list of the C-join messages for PE3 already contains a multicast tunnel interface, which can lead to a potential join loop, because both the upstream and downstream interfaces are multicast tunnel interfaces. In this case, PE1 uses only the EBGP path to send the join messages.

3. If the selected least loaded interface is a multicast tunnel interface and the multicast tunnel interface is not present in the downstream list of the C-join messages, the loop prevention mechanism is not necessary. If any PE router has already advertised data multicast distribution tree (MDT) type, length, and values (TLVs), that PE router is selected as the upstream neighbor.

When the PE1 router sends the join messages to PE2 using the least loaded IBGP path, and if PE3 sends its join messages to PE2, no join loop is created.

4. If no data MDT TLV corresponds to the C-join message, the least loaded neighbor on a multicast tunnel interface is selected as the upstream interface.

On PE routers that have only IBGP paths toward the source (or RP), such as PE3, PIM join load balancing is performed as follows:

1. The PE router only finds a multicast tunnel interface as the RPF interface, and load balancing is done across the C-PIM neighbors on a multicast tunnel interface.

   Router PE3 load-balances PIM join messages received from the CE4 router across the IBGP paths to the PE1 and PE2 routers.

2. If any PE router has already advertised data MDT TLVs corresponding to the C-join messages, that PE router is selected as the RPF neighbor.

For a particular C-multicast flow, at least one of the PE routers having EIBGP paths toward the source (or RP) must use only the EBGP path to avoid or break join loops. As a result of the loop avoidance mechanism, a PE router is constrained to choose among EIBGP paths when a multicast tunnel interface is already present in the downstream list.

In Figure 54 on page 590, assuming that the CE2 host is interested in receiving traffic from the Source and CE2 initiates multiple PIM join messages for different groups (Group 1 with group address 203.0.113.1, and Group 2 with group address 203.0.113.2), the join messages for both groups arrive on the PE1 router.

Router PE1 then equally distributes the join messages between the EIBGP paths toward the Source. Assuming that Group 1 join messages are sent to the CE1 router directly using the EBGP path, and Group 2 join messages are sent to the PE2 router using the IBGP path, PE1 and PE2 become the RPF neighbors for Group 1 and Group 2 join messages, respectively.

When the CE3 router initiates Group 1 and Group 2 PIM join messages, the join messages for both groups arrive on the PE2 router. Router PE2 then equally distributes the join messages between the EIBGP paths toward the Source. Since PE2 is the RPF neighbor for Group 2 join messages, it sends the Group 2 join messages directly to the CE1 router using the EBGP path. Group 1 join messages are sent to the PE1 router using the IBGP path.

However, if the CE4 router initiates multiple Group 1 and Group 2 PIM join messages, there is no control over how these join messages received on the PE3 router get distributed to reach the Source. The selection of the RPF neighbor by PE3 can affect PIM join load balancing on EIBGP paths.

- If PE3 sends Group 1 join messages to PE1 and Group 2 join messages to PE2, there is no change in RPF neighbor. As a result, no join loops are created.

- If PE3 sends Group 1 join messages to PE2 and Group 2 join messages to PE1, there is a change in the RPF neighbor for the different groups resulting in the creation of join loops. To avoid potential join loops, PE1 and PE2 do not consider IBGP paths to send the join messages received from the PE3 router. Instead, the join messages are sent directly to the CE1 router using only the EBGP path.

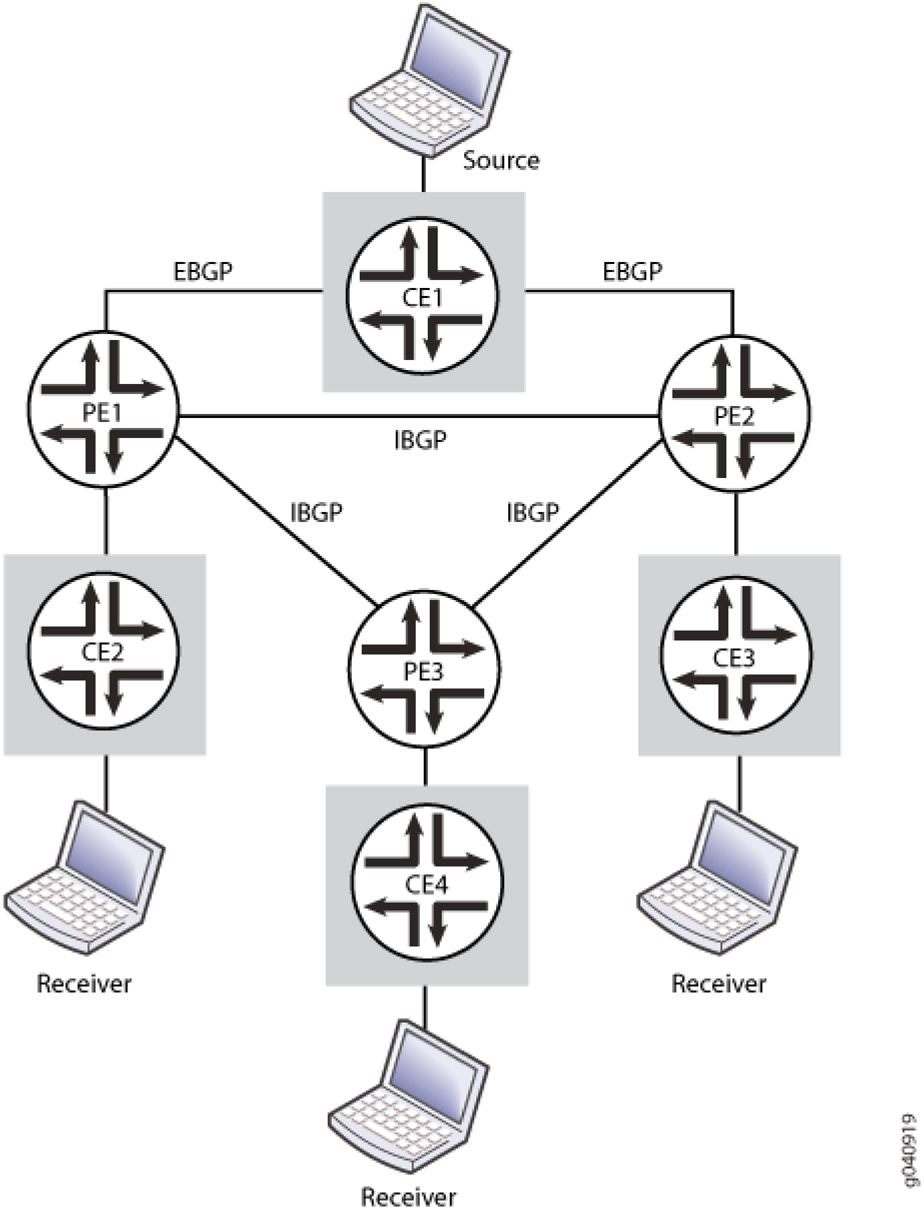
The loop avoidance mechanism in a Draft-Rosen MVPN has the following limitations:

- Because the timing of arrival of join messages on remote PE routers determines the distribution of join messages, the distribution could be sub-optimal in terms of join count.

- Because join loops cannot be avoided and can occur due to the timing of join messages, the subsequent RPF interface change leads to loss of multicast traffic. This can be avoided by implementing the PIM make-before-break feature.

  The PIM make-before-break feature is an approach to detect and break C-PIM join loops in a Draft-Rosen MVPN. The C-PIM join messages are sent to the new RPF neighbor after establishing the PIM neighbor relationship, but before updating the related multicast forwarding entry. Though the upstream RPF neighbor would have updated its multicast forwarding entry and started sending the multicast traffic downstream, the downstream router does not forward the multicast traffic (because of RPF check failure) until the multicast forwarding entry is updated with the new RPF neighbor. This helps to ensure that the multicast traffic is available on the new path before switching the RPF interface of the multicast forwarding entry.

Figure 54: PIM Join Load Balancing on Draft-Rosen MVPN



Configuration

IN THIS SECTION

CLI Quick Configuration

To quickly configure this example, copy the following commands, paste them into a text file, remove any line breaks, change any details necessary to match your network configuration, and then copy and paste the commands into the CLI at the **[edit]** hierarchy level.

PE1

```
set routing-instances vpn1 instance-type vrf
set routing-instances vpn1 interface ge-5/0/4.0
set routing-instances vpn1 interface ge-5/2/0.0
set routing-instances vpn1 interface lo0.1
set routing-instances vpn1 route-distinguisher 1:1
set routing-instances vpn1 vrf-target target:1:1
set routing-instances vpn1 routing-options multipath vpn-unequal-cost equal-external-internal
set routing-instances vpn1 protocols bgp export direct
set routing-instances vpn1 protocols bgp group bgp type external
set routing-instances vpn1 protocols bgp group bgp local-address 192.0.2.4
set routing-instances vpn1 protocols bgp group bgp family inet unicast
set routing-instances vpn1 protocols bgp group bgp neighbor 192.0.2.5 peer-as 3
set routing-instances vpn1 protocols bgp group bgp1 type external
set routing-instances vpn1 protocols bgp group bgp1 local-address 192.0.2.1
set routing-instances vpn1 protocols bgp group bgp1 family inet unicast
set routing-instances vpn1 protocols bgp group bgp1 neighbor 192.0.2.2 peer-as 4
set routing-instances vpn1 protocols pim group-address 198.51.100.1
set routing-instances vpn1 protocols pim rp static address 10.255.8.168
set routing-instances vpn1 protocols pim interface all
set routing-instances vpn1 protocols pim join-load-balance
```

PE2

```
set routing-instances vpn1 instance-type vrf
set routing-instances vpn1 interface ge-2/0/3.0
set routing-instances vpn1 interface ge-4/0/5.0
set routing-instances vpn1 interface lo0.1
set routing-instances vpn1 route-distinguisher 2:2
set routing-instances vpn1 vrf-target target:1:1
set routing-instances vpn1 routing-options multipath vpn-unequal-cost equal-external-internal
set routing-instances vpn1 protocols bgp export direct
set routing-instances vpn1 protocols bgp group bgp1 type external
set routing-instances vpn1 protocols bgp group bgp1 local-address 10.90.10.1
set routing-instances vpn1 protocols bgp group bgp1 family inet unicast
```

```
set routing-instances vpn1 protocols bgp group bgp1 neighbor 10.90.10.2 peer-as 45
set routing-instances vpn1 protocols bgp group bgp type external
set routing-instances vpn1 protocols bgp group bgp local-address 10.50.10.2
set routing-instances vpn1 protocols bgp group bgp family inet unicast
set routing-instances vpn1 protocols bgp group bgp neighbor 10.50.10.1 peer-as 4
set routing-instances vpn1 protocols pim group-address 198.51.100.1
set routing-instances vpn1 protocols pim rp static address 10.255.8.168
set routing-instances vpn1 protocols pim interface all
set routing-instances vpn1 protocols pim join-load-balance
```

Procedure

Step-by-Step Procedure

The following example requires you to navigate various levels in the configuration hierarchy. For information about navigating the CLI, see *Using the CLI Editor in Configuration Mode*. To configure the PE1 router:

> **NOTE**: Repeat this procedure for every Juniper Networks router in the MVPN domain, after modifying the appropriate interface names, addresses, and any other parameters for each router.

1. Configure a VPN routing and forwarding (VRF) instance.

```
[edit routing-instances vpn1]
user@PE1# set instance-type vrf
user@PE1# set interface ge-5/0/4.0
user@PE1# set interface ge-5/2/0.0
user@PE1# set interface lo0.1
user@PE1# set route-distinguisher 1:1
user@PE1# set vrf-target target:1:1
```

2. Enable protocol-independent load balancing for the VRF instance.

```
[edit routing-instances vpn1]
user@PE1# set routing-options multipath vpn-unequal-cost equal-external-internal
```

3. Configure BGP groups and neighbors to enable PE to CE routing.

```
[edit routing-instances vpn1 protocols]
user@PE1# set bgp export direct
user@PE1# set bgp group bgp type external
user@PE1# set bgp group bgp local-address 192.0.2.4
user@PE1# set bgp group bgp family inet unicast
user@PE1# set bgp group bgp neighbor 192.0.2.5 peer-as 3
user@PE1# set bgp group bgp1 type external
user@PE1# set bgp group bgp1 local-address 192.0.2.1
user@PE1# set bgp group bgp1 family inet unicast
user@PE1# set bgp group bgp1 neighbor 192.0.2.2 peer-as 4
```

4. Configure PIM to enable PE to CE multicast routing.

```
[edit routing-instances vpn1 protocols]
user@PE1# set pim group-address 198.51.100.1
user@PE1# set pim rp static address 10.255.8.168
```

5. Enable PIM on all network interfaces.

```
[edit routing-instances vpn1 protocols]
user@PE1# set pim interface all
```

6. Enable PIM join load balancing for the VRF instance.

```
[edit routing-instances vpn1 protocols]
user@PE1# set pim join-load-balance
```

Results

From configuration mode, confirm your configuration by entering the **show routing-instances** command. If the output does not display the intended configuration, repeat the instructions in this example to correct the configuration.

```
routing-instances {
    vpn1 {
        instance-type vrf;
```

```
        interface ge-5/0/4.0;
        interface ge-5/2/0.0;
        interface lo0.1;
        route-distinguisher 1:1;
        vrf-target target:1:1;
        routing-options {
            multipath {
                vpn-unequal-cost equal-external-internal;
            }
        }
        protocols {
            bgp {
                export direct;
                group bgp {
                    type external;
                    local-address 192.0.2.4;
                    family inet {
                        unicast;
                    }
                    neighbor 192.0.2.5 {
                        peer-as 3;
                    }
                }
                group bgp1 {
                    type external;
                    local-address 192.0.2.1;
                    family inet {
                        unicast;
                    }
                    neighbor 192.0.2.2 {
                        peer-as 4;
                    }
                }
            }
            pim {
                group-address 198.51.100.1;
                rp {
                    static {
                        address 10.255.8.168;
                    }
                }
                interface all;
                join-load-balance;
```

```
            }
        }
    }
}
```

If you are done configuring the device, enter **commit** from configuration mode.

Verification

IN THIS SECTION

Confirm that the configuration is working properly.

Verifying PIM Join Load Balancing for Different Groups of Join Messages

Purpose

Verify PIM join load balancing for the different groups of join messages received on the PE1 router.

Action

From operational mode, run the **show pim join instance extensive** command.

```
user@PE1>show pim join instance extensive
 Instance: PIM.vpn1 Family: INET
 R = Rendezvous Point Tree, S = Sparse, W = Wildcard

 Group: 203.0.113.1
     Source: *
     RP: 10.255.8.168
     Flags: sparse,rptree,wildcard
     Upstream interface: ge-5/2/0.1
     Upstream neighbor: 10.10.10.2
     Upstream state: Join to RP
     Downstream neighbors:
         Interface: ge-5/0/4.0
```

```
            10.40.10.2 State: Join Flags: SRW Timeout: 207

Group: 203.0.113.2
    Source: *
    RP: 10.255.8.168
    Flags: sparse,rptree,wildcard
    Upstream interface:  mt-5/0/10.32768
    Upstream neighbor: 19.19.19.19
    Upstream state: Join to RP
    Downstream neighbors:
        Interface: ge-5/0/4.0
            10.40.10.2 State: Join Flags: SRW Timeout: 207

Group: 203.0.113.3
    Source: *
    RP: 10.255.8.168
    Flags: sparse,rptree,wildcard
    Upstream interface: ge-5/2/0.1
    Upstream neighbor: 10.10.10.2
    Upstream state: Join to RP
    Downstream neighbors:
        Interface: ge-5/0/4.0
            10.40.10.2 State: Join Flags: SRW Timeout: 207

Group: 203.0.113.4
    Source: *
    RP: 10.255.8.168
    Flags: sparse,rptree,wildcard
    Upstream interface:  mt-5/0/10.32768
    Upstream neighbor: 19.19.19.19
    Upstream state: Join to RP
    Downstream neighbors:
        Interface: ge-5/0/4.0
            10.40.10.2 State: Join Flags: SRW Timeout: 207
```

Meaning

The output shows how the PE1 router has load-balanced the C-PIM join messages for four different groups.

- For Group 1 (group address: 203.0.113.1) and Group 3 (group address: 203.0.113.3) join messages, the PE1 router has selected the EBGP path toward the CE1 router to send the join messages.

- For Group 2 (group address: 203.0.113.2) and Group 4 (group address: 203.0.113.4) join messages, the PE1 router has selected the IBGP path toward the PE2 router to send the join messages.

SEE ALSO

> PIM Join Load Balancing on Multipath MVPN Routes Overview
>
> Example: Configuring PIM Join Load Balancing on Next-Generation Multicast VPN | **1177**

MBGP Multicast VPN Sites

The main characteristics of MBGP MVPNs are:

- They extend Layer 3 VPN service (RFC 4364) to support IP multicast for Layer 3 VPN service providers.

- They follow the same architecture as specified by RFC 4364 for unicast VPNs. Specifically, BGP is used as the provider edge (PE) router-to-PE router control plane for multicast VPN.

- They eliminate the requirement for the virtual router (VR) model (as specified in Internet draft draft-rosen-vpn-mcast, *Multicast in MPLS/BGP VPNs*) for multicast VPNs and the RFC 4364 model for unicast VPNs.

- They rely on RFC 4364-based unicast with extensions for intra-AS and inter-AS communication.

An MBGP MVPN defines two types of site sets, a sender site set and a receiver site set. These sites have the following properties:

- Hosts within the sender site set can originate multicast traffic for receivers in the receiver site set.

- Receivers outside the receiver site set should not be able to receive this traffic.

- Hosts within the receiver site set can receive multicast traffic originated by any host in the sender site set.

- Hosts within the receiver site set should not be able to receive multicast traffic originated by any host that is not in the sender site set.

A site can be in both the sender site set and the receiver site set, so hosts within such a site can both originate and receive multicast traffic. For example, the sender site set could be the same as the receiver site set, in which case all sites could both originate and receive multicast traffic from one another.

Sites within a given MBGP MVPN might be within the same organization or in different organizations, which means that an MBGP MVPN can be either an intranet or an extranet. A given site can be in more than one MBGP MVPN, so MBGP MVPNs might overlap. Not all sites of a given MBGP MVPN have to be connected to the same service provider, meaning that an MBGP MVPN can span multiple service providers.

Feature parity for the MVPN extranet functionality or overlapping MVPNs on the Junos Trio chipset is supported in Junos OS Releases 11.1R2, 11.2R2, and 11.4.

Another way to look at an MBGP MVPN is to say that an MBGP MVPN is defined by a set of administrative policies. These policies determine both the sender site set and the receiver site set. These policies are established by MBGP MVPN customers, but implemented by service providers using the existing BGP and MPLS VPN infrastructure.

SEE ALSO

Example: Allowing MBGP MVPN Remote Sources
Example: Configuring a PIM-SSM Provider Tunnel for an MBGP MVPN

Example: Configuring MBGP Multicast VPNs

IN THIS SECTION

This example provides a step-by-step procedure to configure multicast services across a multiprotocol BGP (MBGP) Layer 3 virtual private network. (also referred to as next-generation Layer 3 multicast VPNs)

Requirements

This example uses the following hardware and software components:

- Junos OS Release 9.2 or later
- Five M Series, T Series, TX Series, or MX Series Juniper routers
- One host system capable of sending multicast traffic and supporting the Internet Group Management Protocol (IGMP)
- One host system capable of receiving multicast traffic and supporting IGMP

Depending on the devices you are using, you might be required to configure static routes to:

- The multicast sender
- The Fast Ethernet interface to which the sender is connected on the multicast receiver
- The multicast receiver
- The Fast Ethernet interface to which the receiver is connected on the multicast sender

Overview and Topology

IN THIS SECTION

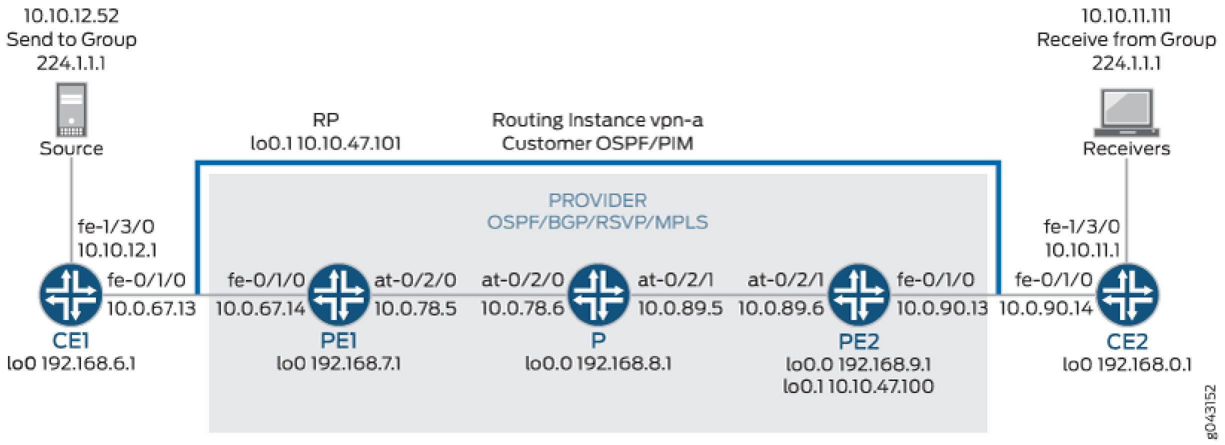
This example shows how to configure the following technologies:

- IPv4
- BGP
- OSPF
- RSVP
- MPLS
- PIM sparse mode
- Static RP

Topology

The topology of the network is shown in Figure 55 on page 600.

Figure 55: Multicast Over Layer 3 VPN Example Topology



Configuration

IN THIS SECTION

NOTE: In any configuration session, it is a good practice to periodically verify that the configuration can be committed using the `commit check` command.

In this example, the router being configured is identified using the following command prompts:

- `CE1` identifies the customer edge 1 (CE1) router
- `PE1` identifies the provider edge 1 (PE1) router
- `P` identifies the provider core (P) router
- `CE2` identifies the customer edge 2 (CE2) router
- `PE2` identifies the provider edge 2 (PE2) router

To configure MBGP multicast VPNs for the network shown in Figure 55 on page 600, perform the following steps:

Configuring Interfaces

Step-by-Step Procedure

The following example requires you to navigate various levels in the configuration hierarchy. For information about navigating the CLI, see *Using the CLI Editor in Configuration Mode* in the CLI User Guide.

1. On each router, configure an IP address on the loopback logical interface 0 (`lo0.0`).

```
[edit interfaces]
user@CE1# set lo0 unit 0 family inet address 192.168.6.1/32 primary

user@PE1# set lo0 unit 0 family inet address 192.168.7.1/32 primary

user@P# set lo0 unit 0 family inet address 192.168.8.1/32 primary

user@PE2# set lo0 unit 0 family inet address 192.168.9.1/32 primary

user@CE2# set lo0 unit 0 family inet address 192.168.0.1/32 primary
```

   Use the `show interfaces terse` command to verify that the IP address is correct on the loopback logical interface.

2. On the PE and CE routers, configure the IP address and protocol family on the Fast Ethernet interfaces. Specify the `inet` protocol family type.

```
[edit interfaces]
user@CE1# set fe-1/3/0 unit 0 family inet address 10.10.12.1/24
user@CE1# set fe-0/1/0 unit 0 family inet address 10.0.67.13/30
```

```
[edit interfaces]
user@PE1# set fe-0/1/0 unit 0 family inet address 10.0.67.14/30

[edit interfaces]
user@PE2# set fe-0/1/0 unit 0 family inet address 10.0.90.13/30

[edit interfaces]
user@CE2# set fe-0/1/0 unit 0 family inet address 10.0.90.14/30
user@CE2# set fe-1/3/0 unit 0 family inet address 10.10.11.1/24
```

Use the `show interfaces terse` command to verify that the IP address is correct on the Fast Ethernet interfaces.

3. On the PE and P routers, configure the ATM interfaces' VPI and maximum virtual circuits. If the default PIC type is different on directly connected ATM interfaces, configure the PIC type to be the same. Configure the logical interface VCI, protocol family, local IP address, and destination IP address.

```
[edit interfaces]
user@PE1# set at-0/2/0 atm-options pic-type atm1
user@PE1# set at-0/2/0 atm-options vpi 0 maximum-vcs 256
user@PE1# set at-0/2/0 unit 0 vci 0.128
user@PE1# set at-0/2/0 unit 0 family inet address 10.0.78.5/32 destination 10.0.78.6

[edit interfaces]
user@P# set at-0/2/0 atm-options pic-type atm1
user@P# set at-0/2/0 atm-options vpi 0 maximum-vcs 256
user@P# set at-0/2/0 unit 0 vci 0.128
user@P# set at-0/2/0 unit 0 family inet address 10.0.78.6/32 destination 10.0.78.5
user@P# set at-0/2/1 atm-options pic-type atm1
user@P# set at-0/2/1 atm-options vpi 0 maximum-vcs 256
user@P# set at-0/2/1 unit 0 vci 0.128
user@P# set at-0/2/1 unit 0 family inet address 10.0.89.5/32 destination 10.0.89.6

[edit interfaces]
user@PE2# set at-0/2/1 atm-options pic-type atm1
user@PE2# set at-0/2/1 atm-options vpi 0 maximum-vcs 256
user@PE2# set at-0/2/1 unit 0 vci 0.128
user@PE2# set at-0/2/1 unit 0 family inet address 10.0.89.6/32 destination 10.0.89.5
```

Use the `show configuration interfaces` command to verify that the ATM interfaces' VPI and maximum VCs are correct and that the logical interface VCI, protocol family, local IP address, and destination IP address are correct.

Configuring OSPF

Step-by-Step Procedure

1. On the P and PE routers, configure the provider instance of OSPF. Specify the `lo0.0` and ATM core-facing logical interfaces. The provider instance of OSPF on the PE router forms adjacencies with the OSPF neighbors on the other PE router and Router P.

   ```
   user@PE1# set protocols ospf area 0.0.0.0 interface at-0/2/0.0
   user@PE1# set protocols ospf area 0.0.0.0 interface lo0.0

   user@P# set protocols ospf area 0.0.0.0 interface lo0.0
   user@P# set protocols ospf area 0.0.0.0 interface all
   user@P# set protocols ospf area 0.0.0.0 interface fxp0 disable

   user@PE2# set protocols ospf area 0.0.0.0 interface lo0.0
   user@PE2# set protocols ospf area 0.0.0.0 interface at-0/2/1.0
   ```

   Use the `show ospf interfaces` command to verify that the `lo0.0` and ATM core-facing logical interfaces are configured for OSPF.

2. On the CE routers, configure the customer instance of OSPF. Specify the loopback and Fast Ethernet logical interfaces. The customer instance of OSPF on the CE routers form adjacencies with the neighbors within the VPN routing instance of OSPF on the PE routers.

   ```
   user@CE1# set protocols ospf area 0.0.0.0 interface fe-0/1/0.0
   user@CE1# set protocols ospf area 0.0.0.0 interface fe-1/3/0.0
   user@CE1# set protocols ospf area 0.0.0.0 interface lo0.0

   user@CE2# set protocols ospf area 0.0.0.0 interface fe-0/1/0.0
   user@CE2# set protocols ospf area 0.0.0.0 interface fe-1/3/0.0
   user@CE2# set protocols ospf area 0.0.0.0 interface lo0.0
   ```

   Use the `show ospf interfaces` command to verify that the correct loopback and Fast Ethernet logical interfaces have been added to the OSPF protocol.

3. On the P and PE routers, configure OSPF traffic engineering support for the provider instance of OSPF.

The `shortcuts` statement enables the master instance of OSPF to use a label-switched path as the next hop.

```
user@PE1# set protocols ospf traffic-engineering shortcuts

user@P# set protocols ospf traffic-engineering shortcuts

user@PE2# set protocols ospf traffic-engineering shortcuts
```

Use the `show ospf overview` or `show configuration protocols ospf` command to verify that traffic engineering support is enabled.

Configuring BGP

Step-by-Step Procedure

1. On Router P, configure BGP for the VPN. The local address is the local `lo0.0` address. The neighbor addresses are the PE routers' `lo0.0` addresses.

   The `unicast` statement enables the router to use BGP to advertise network layer reachability information (NLRI). The `signaling` statement enables the router to use BGP as the signaling protocol for the VPN.

   ```
   user@P# set protocols bgp group group-mvpn type internal
   user@P# set protocols bgp group group-mvpn local-address 192.168.8.1
   user@P# set protocols bgp group group-mvpn family inet unicast
   user@P# set protocols bgp group group-mvpn family inet-mvpn signaling
   user@P# set protocols bgp group group-mvpn neighbor 192.168.9.1
   user@P# set protocols bgp group group-mvpn neighbor 192.168.7.1
   ```

   Use the `show configuration protocols bgp` command to verify that the router has been configured to use BGP to advertise NLRI.

2. On the PE and P routers, configure the BGP local autonomous system number.

   ```
   user@PE1# set routing-options autonomous-system 0.65010

   user@P# set routing-options autonomous-system 0.65010

   user@PE2# set routing-options autonomous-system 0.65010
   ```

Use the `show configuration routing-options` command to verify that the BGP local autonomous system number is correct.

3. On the PE routers, configure BGP for the VPN. Configure the local address as the local `lo0.0` address. The neighbor addresses are the `lo0.0` addresses of Router P and the other PE router, PE2.

```
user@PE1# set protocols bgp group group-mvpn type internal
user@PE1# set protocols bgp group group-mvpn local-address 192.168.7.1
user@PE1# set protocols bgp group group-mvpn family inet-vpn unicast
user@PE1# set protocols bgp group group-mvpn family inet-mvpn signaling
user@PE1# set protocols bgp group group-mvpn neighbor 192.168.9.1
user@PE1# set protocols bgp group group-mvpn neighbor 192.168.8.1

user@PE2# set protocols bgp group group-mvpn type internal
user@PE2# set protocols bgp group group-mvpn local-address 192.168.9.1
user@PE2# set protocols bgp group group-mvpn family inet-vpn unicast
user@PE2# set protocols bgp group group-mvpn family inet-mvpn signaling
user@PE2# set protocols bgp group group-mvpn neighbor 192.168.7.1
user@PE2# set protocols bgp group group-mvpn neighbor 192.168.8.1
```

Use the `show bgp group` command to verify that the BGP configuration is correct.

4. On the PE routers, configure a policy to export the BGP routes into OSPF.

```
user@PE1# set policy-options policy-statement bgp-to-ospf from protocol bgp
user@PE1# set policy-options policy-statement bgp-to-ospf then accept

user@PE2# set policy-options policy-statement bgp-to-ospf from protocol bgp
user@PE2# set policy-options policy-statement bgp-to-ospf then accept
```

Use the `show policy bgp-to-ospf` command to verify that the policy is correct.

Configuring RSVP

Step-by-Step Procedure

1. On the PE routers, enable RSVP on the interfaces that participate in the LSP. Configure the Fast Ethernet and ATM logical interfaces.

```
user@PE1# set protocols rsvp interface fe-0/1/0.0
user@PE1# set protocols rsvp interface at-0/2/0.0
```

```
user@PE2# set protocols rsvp interface fe-0/1/0.0
user@PE2# set protocols rsvp interface at-0/2/1.0
```

2. On Router P, enable RSVP on the interfaces that participate in the LSP. Configure the ATM logical interfaces.

```
user@P# set protocols rsvp interface at-0/2/0.0
user@P# set protocols rsvp interface at-0/2/1.0
```

   Use the `show configuration protocols rsvp` command to verify that the RSVP configuration is correct.

Configuring MPLS

Step-by-Step Procedure

1. On the PE routers, configure an MPLS LSP to the PE router that is the LSP egress point. Specify the IP address of the `lo0.0` interface on the router at the other end of the LSP. Configure MPLS on the ATM, Fast Ethernet, and `lo0.0` interfaces.

   To help identify each LSP when troubleshooting, configure a different LSP name on each PE router. In this example, we use the name `to-pe2` as the name for the LSP configured on PE1 and `to-pe1` as the name for the LSP configured on PE2.

```
user@PE1# set protocols mpls label-switched-path to-pe2 to 192.168.9.1
user@PE1# set protocols mpls interface fe-0/1/0.0
user@PE1# set protocols mpls interface at-0/2/0.0
user@PE1# set protocols mpls interface lo0.0

user@PE2# set protocols mpls label-switched-path to-pe1 to 192.168.7.1
user@PE2# set protocols mpls interface fe-0/1/0.0
user@PE2# set protocols mpls interface at-0/2/1.0
user@PE2# set protocols mpls interface lo0.0
```

   Use the `show configuration protocols mpls` and `show route label-switched-path to-pe1` commands to verify that the MPLS and LSP configuration is correct.

   After the configuration is committed, use the `show mpls lsp name to-pe1` and `show mpls lsp name to-pe2` commands to verify that the LSP is operational.

2. On Router P, enable MPLS. Specify the ATM interfaces connected to the PE routers.

```
user@P# set protocols mpls interface at-0/2/0.0
user@P# set protocols mpls interface at-0/2/1.0
```

   Use the `show mpls interface` command to verify that MPLS is enabled on the ATM interfaces.

3. On the PE and P routers, configure the protocol family on the ATM interfaces associated with the LSP. Specify the `mpls` protocol family type.

```
user@PE1# set interfaces at-0/2/0 unit 0 family mpls

user@P# set interfaces at-0/2/0 unit 0 family mpls
user@P# set interfaces at-0/2/1 unit 0 family mpls

user@PE2# set interfaces at-0/2/1 unit 0 family mpls
```

   Use the `show mpls interface` command to verify that the MPLS protocol family is enabled on the ATM interfaces associated with the LSP.

Configuring the VRF Routing Instance

Step-by-Step Procedure

1. On the PE routers, configure a routing instance for the VPN and specify the `vrf` instance type. Add the Fast Ethernet and `lo0.1` customer-facing interfaces. Configure the VPN instance of OSPF and include the BGP-to-OSPF export policy.

```
user@PE1# set routing-instances vpn-a instance-type vrf
user@PE1# set routing-instances vpn-a interface lo0.1
user@PE1# set routing-instances vpn-a interface fe-0/1/0.0
user@PE1# set routing-instances vpn-a protocols ospf export bgp-to-ospf
user@PE1# set routing-instances vpn-a protocols ospf area 0.0.0.0 interface all

user@PE2# set routing-instances vpn-a instance-type vrf
user@PE2# set routing-instances vpn-a interface lo0.1
user@PE2# set routing-instances vpn-a interface fe-0/1/0.0
user@PE2# set routing-instances vpn-a protocols ospf export bgp-to-ospf
user@PE2# set routing-instances vpn-a protocols ospf area 0.0.0.0 interface all
```

Use the `show configuration routing-instances vpn-a` command to verify that the routing instance configuration is correct.

2. On the PE routers, configure a route distinguisher for the routing instance. A route distinguisher allows the router to distinguish between two identical IP prefixes used as VPN routes. Configure a different route distinguisher on each PE router. This example uses 65010:1 on PE1 and 65010:2 on PE2.

```
user@PE1# set routing-instances vpn-a route-distinguisher 65010:1

user@PE2# set routing-instances vpn-a route-distinguisher 65010:2
```

Use the `show configuration routing-instances vpn-a` command to verify that the route distinguisher is correct.

3. On the PE routers, configure default VRF import and export policies. Based on this configuration, BGP automatically generates local routes corresponding to the route target referenced in the VRF import policies. This example uses 2:1 as the route target.

> **NOTE**: You must configure the same route target on each PE router for a given VPN routing instance.

```
user@PE1# set routing-instances vpn-a vrf-target target:2:1

user@PE2# set routing-instances vpn-a vrf-target target:2:1
```

Use the `show configuration routing-instances vpn-a` command to verify that the route target is correct.

4. On the PE routers, configure the VPN routing instance for multicast support.

```
user@PE1# set routing-instances vpn-a protocols mvpn

user@PE2# set routing-instances vpn-a protocols mvpn
```

Use the `show configuration routing-instance vpn-a` command to verify that the VPN routing instance has been configured for multicast support.

5. On the PE routers, configure an IP address on loopback logical interface 1 (`lo0.1`) used in the customer routing instance VPN.

   ```
   user@PE1# set interfaces lo0 unit 1 family inet address 10.10.47.101/32

   user@PE2# set interfaces lo0 unit 1 family inet address 10.10.47.100/32
   ```

   Use the `show interfaces terse` command to verify that the IP address on the loopback interface is correct.

Configuring PIM

Step-by-Step Procedure

1. On the PE routers, enable PIM. Configure the `lo0.1` and the customer-facing Fast Ethernet interface. Specify the mode as `sparse` and the version as `2`.

   ```
   user@PE1# set routing-instances vpn-a protocols pim interface lo0.1 mode sparse
   user@PE1# set routing-instances vpn-a protocols pim interface lo0.1 version 2
   user@PE1# set routing-instances vpn-a protocols pim interface fe-0/1/0.0 mode sparse
   user@PE1# set routing-instances vpn-a protocols pim interface fe-0/1/0.0 version 2
   user@PE2# set routing-instances vpn-a protocols pim interface lo0.1 mode sparse
   user@PE2# set routing-instances vpn-a protocols pim interface lo0.1 version 2
   user@PE2# set routing-instances vpn-a protocols pim interface fe-0/1/0.0 mode sparse
   user@PE2# set routing-instances vpn-a protocols pim interface fe-0/1/0.0 version 2
   ```

   Use the `show pim interfaces instance vpn-a` command to verify that PIM sparse-mode is enabled on the `lo0.1` interface and the customer-facing Fast Ethernet interface.

2. On the CE routers, enable PIM. In this example, we configure all interfaces. Specify the mode as `sparse` and the version as `2`.

   ```
   user@CE1# set protocols pim interface all
   user@CE2# set protocols pim interface all mode sparse
   user@CE2# set protocols pim interface all version 2
   ```

   Use the `show pim interfaces` command to verify that PIM sparse mode is enabled on all interfaces.

Configuring the Provider Tunnel

Step-by-Step Procedure

1. On Router PE1, configure the provider tunnel. Specify the multicast address to be used.

   The `provider-tunnel` statement instructs the router to send multicast traffic across a tunnel.

   ```
   user@PE1# set routing-instances vpn-a provider-tunnel rsvp-te label-switched-path-template
   default-template
   ```

   Use the `show configuration routing-instance vpn-a` command to verify that the provider tunnel is configured to use the default LSP template.

2. On Router PE2, configure the provider tunnel. Specify the multicast address to be used.

   ```
   user@PE2# set routing-instances vpn-a provider-tunnel rsvp-te label-switched-path-template
   default-template
   ```

   Use the `show configuration routing-instance vpn-a` command to verify that the provider tunnel is configured to use the default LSP template.

Configuring the Rendezvous Point

Step-by-Step Procedure

1. Configure Router PE1 to be the rendezvous point. Specify the `lo0.1` address of Router PE1. Specify the multicast address to be used.

   ```
   user@PE1# set routing-instances vpn-a protocols pim rp local address 10.10.47.101
   user@PE1# set routing-instances vpn-a protocols pim rp local group-ranges 224.1.1.1/32
   ```

   Use the `show pim rps instance vpn-a` command to verify that the correct local IP address is configured for the RP.

2. On Router PE2, configure the static rendezvous point. Specify the `lo0.1` address of Router PE1.

   ```
   user@PE2# set routing-instances vpn-a protocols pim rp static address 10.10.47.101
   ```

Use the `show pim rps instance vpn-a` command to verify that the correct static IP address is configured for the RP.

3. On the CE routers, configure the static rendezvous point. Specify the `lo0.1` address of Router PE1.

   ```
   user@CE1# set protocols pim rp static address 10.10.47.101 version 2
   user@CE2# set protocols pim rp static address 10.10.47.101 version 2
   ```

   Use the `show pim rps` command to verify that the correct static IP address is configured for the RP.

4. Use the `commit check` command to verify that the configuration can be successfully committed. If the configuration passes the check, commit the configuration.

5. Start the multicast sender device connected to CE1.

6. Start the multicast receiver device connected to CE2.

7. Verify that the receiver is receiving the multicast stream.

8. Use `show` commands to verify the routing, VPN, and multicast operation.

Results

The configuration and verification parts of this example have been completed. The following section is for your reference.

The relevant sample configuration for Router CE1 follows.

Router CE1

```
interfaces {
    lo0 {
        unit 0 {
            family inet {
                address 192.168.6.1/32 {
                    primary;
                }
            }
        }
    }
    fe-0/1/0 {
        unit 0 {
            family inet {
                address 10.0.67.13/30;
```

```
            }
        }
    }
    fe-1/3/0 {
        unit 0 {
            family inet {
                address 10.10.12.1/24;
            }
        }
    }
}
protocols {
    ospf {
        area 0.0.0.0 {
            interface fe-0/1/0.0;
            interface lo0.0;
            interface fe-1/3/0.0;
        }
    }
    pim {
        rp {
            static {
                address 10.10.47.101 {
                    version 2;
                }
            }
        }
        interface all;
    }
}
```

The relevant sample configuration for Router PE1 follows.

Router PE1

```
interfaces {
    lo0 {
        unit 0 {
            family inet {
                address 192.168.7.1/32 {
                    primary;
                }
```

```
            }
        }
    }
    fe-0/1/0 {
        unit 0 {
            family inet {
                address 10.0.67.14/30;
            }
        }
    }
    at-0/2/0 {
        atm-options {
            pic-type atm1;
            vpi 0 {
                maximum-vcs 256;
            }
        }
        unit 0 {
            vci 0.128;
            family inet {
                address 10.0.78.5/32 {
                    destination 10.0.78.6;
                }
            }
            family mpls;
        }
    }
    lo0 {
        unit 1 {
            family inet {
                address 10.10.47.101/32;
            }
        }
    }
}
routing-options {
    autonomous-system 0.65010;
}
protocols {
    rsvp {
        interface fe-0/1/0.0;
        interface at-0/2/0.0;
    }
```

```
    mpls {
        label-switched-path to-pe2 {
            to 192.168.9.1;
        }
        interface fe-0/1/0.0;
        interface at-0/2/0.0;
        interface lo0.0;
    }
    bgp {
        group group-mvpn {
            type internal;
            local-address 192.168.7.1;
            family inet-vpn {
                unicast;
            }
            family inet-mvpn {
                signaling;
            }
            neighbor 192.168.9.1;
            neighbor 192.168.8.1;
        }
    }
    ospf {
        traffic-engineering {
            shortcuts;
        }
        area 0.0.0.0 {
            interface at-0/2/0.0;
            interface lo0.0;
        }
    }
}
policy-options {
    policy-statement bgp-to-ospf {
        from protocol bgp;
        then accept;
    }
}
routing-instances {
    vpn-a {
        instance-type vrf;
        interface lo0.1;
        interface fe-0/1/0.0;
```

```
            route-distinguisher 65010:1;
            provider-tunnel {
                rsvp-te {
                    label-switched-path-template {
                        default-template;
                    }
                }
            }
            vrf-target target:2:1;
            protocols {
                ospf {
                    export bgp-to-ospf;
                    area 0.0.0.0 {
                        interface all;
                    }
                }
                pim {
                    rp {
                        local {
                            address 10.10.47.101;
                            group-ranges {
                                224.1.1.1/32;
                            }
                        }
                    }
                    interface lo0.1 {
                        mode sparse;
                        version 2;
                    }
                    interface fe-0/1/0.0 {
                        mode sparse;
                        version 2;
                    }
                }
                mvpn;
            }
        }
    }
```

The relevant sample configuration for Router P follows.

Router P

```
interfaces {
    lo0 {
        unit 0 {
            family inet {
                address 192.168.8.1/32 {
                    primary;
                }
            }
        }
    }
    at-0/2/0 {
        atm-options {
            pic-type atm1;
            vpi 0 {
                maximum-vcs 256;
            }
        }
        unit 0 {
            vci 0.128;
            family inet {
                address 10.0.78.6/32 {
                    destination 10.0.78.5;
                }
            }
            family mpls;
        }
    }
    at-0/2/1 {
        atm-options {
            pic-type atm1;
            vpi 0 {
                maximum-vcs 256;
            }
        }
        unit 0 {
            vci 0.128;
            family inet {
                address 10.0.89.5/32 {
                    destination 10.0.89.6;
                }
```

```
            }
            family mpls;
        }
    }
}
routing-options {
    autonomous-system 0.65010;
}
protocols {
    rsvp {
        interface at-0/2/0.0;
        interface at-0/2/1.0;
    }
    mpls {
        interface at-0/2/0.0;
        interface at-0/2/1.0;
    }
    bgp {
        group group-mvpn {
            type internal;
            local-address 192.168.8.1;
            family inet {
                unicast;
            }
            family inet-mvpn {
                signaling;
            }
            neighbor 192.168.9.1;
            neighbor 192.168.7.1;
        }
    }
    ospf {
        traffic-engineering {
            shortcuts;
        }
        area 0.0.0.0 {
            interface lo0.0;
            interface all;
            interface fxp0.0 {
                disable;
            }
        }
```

```
    }
}
```

The relevant sample configuration for Router PE2 follows.

Router PE2

```
interfaces {
    lo0 {
        unit 0 {
            family inet {
                address 192.168.9.1/32 {
                    primary;
                }
            }
        }
    }
    fe-0/1/0 {
        unit 0 {
            family inet {
                address 10.0.90.13/30;
            }
        }
    }
    at-0/2/1 {
        atm-options {
            pic-type atm1;
            vpi 0 {
                maximum-vcs 256;
            }
        }
        unit 0 {
            vci 0.128;
            family inet {
                address 10.0.89.6/32 {
                    destination 10.0.89.5;
                }
            }
            family mpls;
        }
    }
    lo0 {
```

```
        unit 1 {
            family inet {
                address 10.10.47.100/32;
            }
        }
    }
}
routing-options {
    autonomous-system 0.65010;
}
protocols {
    rsvp {
        interface fe-0/1/0.0;
        interface at-0/2/1.0;
    }
    mpls {
        label-switched-path to-pe1 {
            to 192.168.7.1;
        }
        interface lo0.0;
        interface fe-0/1/0.0;
        interface at-0/2/1.0;
    }
    bgp {
        group group-mvpn {
            type internal;
            local-address 192.168.9.1;
            family inet-vpn {
                unicast;
            }
            family inet-mvpn {
                signaling;
            }
            neighbor 192.168.7.1;
            neighbor 192.168.8.1;
        }
    }
    ospf {
        traffic-engineering {
            shortcuts;
        }
        area 0.0.0.0 {
            interface lo0.0;
```

```
            interface at-0/2/1.0;
        }
    }
}
policy-options {
    policy-statement bgp-to-ospf {
        from protocol bgp;
        then accept;
    }
}
routing-instances {
    vpn-a {
        instance-type vrf;
        interface fe-0/1/0.0;
        interface lo0.1;
        route-distinguisher 65010:2;
        provider-tunnel {
            rsvp-te {
                label-switched-path-template {
                    default-template;
                }
            }
        }
        vrf-target target:2:1;
        protocols {
            ospf {
                export bgp-to-ospf;
                area 0.0.0.0 {
                    interface all;
                }
            }
            pim {
                rp {
                    static {
                        address 10.10.47.101;
                    }
                }
                interface fe-0/1/0.0 {
                    mode sparse;
                    version 2;
                }
                interface lo0.1 {
                    mode sparse;
```

```
                    version 2;
                }
            }
            mvpn;
        }
    }
}
```

The relevant sample configuration for Router CE2 follows.

Router CE2

```
interfaces {
    lo0 {
        unit 0 {
            family inet {
                address 192.168.0.1/32 {
                    primary;
                }
            }
        }
    }
    fe-0/1/0 {
        unit 0 {
            family inet {
                address 10.0.90.14/30;
            }
        }
    }
    fe-1/3/0 {
        unit 0 {
            family inet {
                address 10.10.11.1/24;
            }
            family inet6 {
                address fe80::205:85ff:fe88:ccdb/64;
            }
        }
    }
}
protocols {
    ospf {
```

```
        area 0.0.0.0 {
            interface fe-0/1/0.0;
            interface lo0.0;
            interface fe-1/3/0.0;
        }
    }
    pim {
        rp {
            static {
                address 10.10.47.101 {
                    version 2;
                }
            }
        }
        interface all {
            mode sparse;
            version 2;
        }
    }
}
```

Configuring Point-to-Multipoint LSPs for an MBGP MVPN

IN THIS SECTION

The Junos OS supports point-to-multipoint label-switched paths (LSPs) for MBGP MVPNs. Point-to-multipoint LSPs for multicast VPNs are supported for intra-autonomous system (AS) environments (within an AS), but are not supported for inter-AS environments (between autonomous systems). A point-to-multipoint LSP is an RSVP-signaled LSP with a single source and multiple destinations.

You can configure point-to-multipoint LSPs for MBGP MVPNs as follows:

- Static point-to-multipoint LSPs—Configure static point-to-multipoint LSPs using the standard MPLS LSP statements specified at the `[edit protocols mpls]` hierarchy level. You manually configure each of the leaf nodes for the point-to-multipoint LSP.

- Dynamic point-to-multipoint LSPs using the default template—Configuring dynamic point-to-multipoint LSPs using the `default-template` option causes the leaf nodes to be discovered automatically. The leaf nodes are discovered through BGP intra-AS automatic discovery. The `default-template` option allows you to minimize the amount of configuration needed. However, it does not allow you to configure any of the standard MPLS options.

- Dynamic point-to-multipoint LSPs using a user-configured template—Configuring dynamic point-to-multipoint LSPs using a user-configured template also causes the leaf nodes to be discovered automatically. By creating your own template for the point-to-multipoint LSPs, all of the standard MPLS features (such as bandwidth allocation and traffic engineering) can be configured.

Be aware of the following properties for the egress PE router in a point-to-multipoint LSP configured for a multicast VPN:

- Penultimate hop-popping is not used by point-to-multipoint LSPs for multicast VPNs. Only ultimate hop-popping is used.

- You must configure either the `vrf-table-label` statement or a virtual loopback tunnel interface on the egress PE router.

- If you configure the `vrf-table-label` statement on the egress PE router, and the egress PE router is also a transit router for the point-to-multipoint LSP, the penultimate hop router sends two copies of each packet over the link to the egress PE router.

- If you configure the `vrf-table-label` statement on the egress PE router, and the egress PE router is not a transit router for the point-to-multipoint LSP, the penultimate hop router can send just one copy of each packet over the link to the egress PE router.

- If you configure a virtual loopback tunnel interface on the egress PE router, and the egress PE router is also a transit router for the point-to-multipoint LSP, the penultimate hop router sends just one copy of each packet over the link to the egress PE router. A virtual loopback tunnel interface can perform two lookups on an incoming packet, one for the multicast MPLS lookup and one for the IP lookup.

NOTE: Junos OS Release 11.2 and earlier do not support point-to-multipoint LSPs with next-generation multicast VPNs on MX80 routers.

The following sections describe how to configure point-to-multipoint LSPs for MBGP MVPNs:

Configuring RSVP-Signaled Inclusive Point-to-Multipoint LSPs for an MBGP MVPN

You can configure LDP-signaled or RSVP-signaled inclusive point-to-multipoint LSPs for MBGP MVPNs. Aggregation is not supported, so you need to configure an inclusive point-to-multipoint LSP for each sender PE router in each multicast VPN routing instance. The sender PE router is in the sender site set of the MBGP MVPN.

To configure a static RSVP-signaled inclusive point-to-multipoint LSP, include the `static-lsp` statement:

```
static-lsp lsp-name;
```

You can include this statement at the following hierarchy levels:

- `[edit routing-instances routing-instance-name provider-tunnel rsvp-te]`
- `[edit logical-systems logical-system-name routing-instances routing-instance-name provider-tunnel rsvp-te]`

To configure dynamic inclusive point-to-multipoint LSPs, include the `label-switched-path-template` statement:

```
label-switched-path-template (Multicast) {
    (default-template | lsp-template-name);
}
```

You can include this statement at the following hierarchy levels:

- `[edit routing-instances routing-instance-name provider-tunnel rsvp-te]`
- `[edit logical-systems logical-system-name routing-instances routing-instance-name provider-tunnel rsvp-te]`

You can configure either the `default-template` option or manually configure a point-to-multipoint LSP template and specify the template name.

Configuring Selective Provider Tunnels for an MBGP MVPN

You can configure LDP-signaled or RSVP-signaled selective point-to-multipoint LSPs (also referred to as selective provider tunnels) for MBGP MVPNs. Selective point-to-multipoint LSPs send traffic only to the receivers configured for the multicast VPNs, helping to minimize flooding in the service provider's network.

As with inclusive point-to-multipoint LSPs, you can configure both dynamic and static selective tunnels for the multicast VPN.

To configure selective point-to-multipoint provider tunnels, include the `selective` statement:

```
selective {
    group multicast--prefix/prefix-length {
        source ip--prefix/prefix-length {
```

```
        ldp-p2mp;
        pim-ssm {
            group-range multicast-prefix;
        }
        rsvp-te {
            label-switched-path-template {
                (default-template | lsp-template-name);
            }
            static-lsp point-to-multipoint-lsp-name;
        }
        threshold-rate kbps;
    }
    wildcard-source {
        ldp-p2mp;
        pim-ssm {
            group-range multicast-prefix;
        }
        rsvp-te {
            label-switched-path-template {
                (default-template | lsp-template-name);
            }
            static-lsp point-to-multipoint-lsp-name;
        }
        threshold-rate kbps;
    }
}
tunnel-limit number;
wildcard-group-inet {
    wildcard-source {
        ldp-p2mp;
        pim-ssm {
            group-range multicast-prefix;
        }
        rsvp-te {
            label-switched-path-template {
                (default-template | lsp-template-name);
            }
            static-lsp lsp-name;
        }
        threshold-rate number;
    }
}
wildcard-group-inet6 {
```

```
        wildcard-source {
            ldp-p2mp;
            pim-ssm {
                group-range multicast-prefix;
            }
            rsvp-te {
                label-switched-path-template {
                    (default-template | lsp-template-name);
                }
                static-lsp lsp-name;
            }
            threshold-rate number;
        }
    }
}
```

You can include these statements at the following hierarchy levels:

- `[edit routing-instances routing-instance-name provider-tunnel]`
- `[edit logical-systems logical-system-name routing-instances routing-instance-name provider-tunnel]`

The following sections describe how to configure selective point-to-multipoint LSPs for MBGP MVPNs:

Configuring the Multicast Group Address for an MBGP MVPN

To configure a point-to-multipoint LSP for an MBGP MVPN, you need to specify a multicast group address by including the `group` statement:

```
group address { ... }
```

You can include this statements at the following hierarchy levels:

- `[edit routing-instances routing-instance-name provider-tunnel selective]`
- `[edit logical-systems logical-system-name routing-instances routing-instance-name provider-tunnel selective]`

The address must be a valid multicast group address. Multicast uses the Class D IP address range (`224.0.0.0` through `239.255.255.255`).

Configuring the Multicast Source Address for an MBGP MVPN

To configure a point-to-multipoint LSP for an MBGP MVPN, specify a multicast source address by including the `source` statement:

```
source address { ... }
```

You can include this statement at the following hierarchy levels:

- `[edit routing-instances routing-instance-name provider-tunnel selective group address]`
- `[edit logical-systems logical-system-name routing-instances routing-instance-name provider-tunnel selective group address]`

Configuring Static Selective Point-to-Multipoint LSPs for an MBGP MVPN

You can configure a static selective point-to-multipoint LSP for an MBGP MVPN. You need to configure a static LSP using the standard MPLS LSP statements at the `[edit protocols mpls]` hierarchy level. You then include the static LSP in your selective point-to-multipoint LSP configuration by using the `static-lsp` statement. Once this functionality is enabled on the source PE router, the static point-to-multipoint LSP is created based on your configuration.

To configure a static selective point-to-multipoint LSP, include the `rsvp-te` and the `static-lsp` statements:

```
rsvp-te static-lsp lsp-name;
```

You can include these statements at the following hierarchy levels:

- `[edit routing-instances routing-instance-name provider-tunnel selective group address source source-address]`
- `[edit logical-systems logical-system-name routing-instances routing-instance-name provider-tunnel selective group address source source-address]`

Configuring Dynamic Selective Point-to-Multipoint LSPs for an MBGP MVPN

You can configure a dynamic selective point-to-multipoint LSP for an MBGP MVPN. The leaf nodes for a dynamic point-to-multipoint LSP can be automatically discovered using leaf automatic discovery routes. Selective provider multicast service interface (S-PMSI) automatic discovery routes are also supported.

To configure a dynamic selective point-to-multipoint provider tunnel, include the `rsvp-te` and `label-switched-path-template` statements:

```
rsvp-te label-switched-path-template {
    (default-template | lsp-template-name);
}
```

You can include these statements at the following hierarchy levels:

- `[edit routing-instances` *`routing-instance-name`* `provider-tunnel selective group` *`address`* `source` *`source-address`*`]`
- `[edit logical-systems` *`logical-system-name`* `routing-instances` *`routing-instance-name`* `provider-tunnel selective group` *`address`* `source` *`source-address`*`]`

The `label-switched-path-template` statement includes the following options:

- `default-template`—Specify that point-to-multipoint LSPs are generated dynamically based on the default template. No user configuration is required for the LSPs. However, the automatically generated LSPs include none of the common LSP features, such as bandwidth allocation and traffic engineering.
- *`lsp-template-name`*—Specify the name of an LSP template to be used for the point-to-multipoint LSP. You need to configure the LSP template to be used as a basis for the point-to-multipoint LSPs. You can configure any of the common LSP features for this template.

Configuring the Threshold for Dynamic Selective Point-to-Multipoint LSPs for an MBGP MVPN

To configure a selective point-to-multipoint LSP dynamically, you need to specify the data threshold (in kilobits per second) required before a new tunnel is created using the `threshold-rate` statement:

```
threshold-rate number;
```

You can include this statement at the following hierarchy levels:

- `[edit routing-instances` *`routing-instance-name`* `provider-tunnel selective group` *`address`* `source` *`source-address`*`]`
- `[edit logical-systems` *`logical-system-name`* `routing-instances` *`routing-instance-name`* `provider-tunnel selective group` *`address`* `source` *`source-address`*`]`

Configuring the Tunnel Limit for Dynamic Selective Point-to-Multipoint LSPs for an MBGP MVPN

To configure a limit on the number of tunnels that can be generated for a dynamic point-to-multipoint LSP, include the `tunnel-limit` statement:

```
tunnel-limit number;
```

You can include this statement at the following hierarchy levels:

- `[edit routing-instances routing-instance-name provider-tunnel selective]`
- `[edit logical-systems logical-system-name routing-instances routing-instance-name provider-tunnel selective]`

SEE ALSO

> Example: Configuring Point-to-Multipoint LDP LSPs as the Data Plane for Intra-AS MBGP MVPNs

Segmented Inter-Area Point-to-Multipoint Label-Switched Paths Overview

Junos OS supports point-to-multipoint (P2MP) label-switched paths (LSPs) for BGP MVPNs. BGP MVPN supports non-segmented intra-autonomous systems (ASs) and segmented inter-autonomous systems (ASs).

In order to connect PE routers that are in different areas but in the same AS and require P2MP connectivity, Junos OS allows you to segment the P2MP LSPs at the area boundary as described in Internet draft *draft-ietf-mpls-seamless-mcast-14.txt*. You can use non-segmented LSPs for low-rate multicast flows, and segmented LSPs for high-rate flows. A segmented P2MP LSP within an AS consists of the following segments:

- Ingress area segment — The ingress area segment is rooted at a PE router or autonomous system boundary router (ASBR). The leaves of this segment are PEs, ASBRs, or area border routers (ABRs).
- Backbone area segment — The backbone area segment is rooted at an ABR that is connected to the ingress area/ingress ABR.
- Egress area segment — The egress area segment is rooted at an ABR in the egress area or egress ABR.

NOTE: These areas can be IGP areas or areas based on BGP peer groups, where ABR can be a region border router (RBR). In either case, the transit ABRs/RBRs should be configured on the BGP route reflector (RR).

Each of the intra-area segments can be carried over provider tunnels such as P2MP RSVP-TE LSP, P2MP mLDP LSP, or ingress replication.

Segmentation of inter-area P2MP LSP occurs when the S-PMSI autodiscovery (AD) routes are advertised. This triggers the inclusion of a new BGP extended community or inter-area P2MP segmented next-hop extended community. The segmented inter-area P2MP LSP can be separated into the following three different roles:

- Ingress PE or ASBR — Ingress PE router originates S-PMSI A-D routes. If inter-region segmentation is required, then the PE router generates the S-PMSI A-D routes carrying the inter-area P2MP segmented next-hop router (S-NH) community. The inter-region segmentation can be added for any selective tunnel. The segmentation can happen based on the threshold or fan-out attributes. If the threshold is configured for a selective tunnel, then MVPN starts migrating the flow to a segmented S-PMSI on reaching the threshold rate value. The threshold attribute applies to RSVP, LDP, and IR tunnels. You can trigger the segmentation based on the fan-out attribute, which is the number of leaves. Once the number of leaf A-D routes exceeds the fan-out value, the traffic flow is moved to segmented S-PMSI. The fan-out attribute for LDP tunnels is not applicable at the ingress PE router. If the S-PMSI with ingress replication has configured only the threshold, then the threshold is used to trigger the migration to segmented LSP. If fan-out is also set, then the migration is triggered when the traffic rate multiplied by the number of leaf A-D routes exceeds the threshold value. The segmented threshold and fan-out values are checked based on the existing data threshold checking interval, which by default is every 60 seconds. This prevents the flow from getting migrated too frequently.

- Transit ABRs — When the transit ABR (either ingress ABR or egress ABR) receives an S-PMSI A-D route with the segmentation of inter-region configured, the ABR checks if the S-PMSI is carrying a S-NH extended community attribute. If the S-NH attribute is present in the incoming S-PMSI, then the ABR checks for the tunnel-type to be carried by the S-PMSI. The ABR then generates the tunnel-type across the backbone area or the egress area .

  NOTE: An ABR can set a template to define the provider tunnel type in each region or BGP group. The tunnel type in each region can be incoming, ingress-replication, LDP-P2MP, or RSVP-TE.

  If the tunnel type is incoming, then it indicates that the tunnel type across the ABR remains the same. If the tunnel type is different across the ABR, then the transit ABR modifies the S-PMSI tunnel attribute and the S-NH attribute to its router-id and re-advertises the route to its BGP peers. If no template is configured on the ABR then the ABR simply reflects the incoming S-PMSI routes without changing any of the attributes to its BGP peers.

- Egress PE or ASBR — Egress PE routers or ASBRs learn the upstream node from the segmented next-hop extended community carried in the received S-PMSI A-D routes and responds with the leaf A-D routes carrying the upstream node IP address in the route target extended community (EC).

You can configure the BGP policy to accept or reject the S-PMSI A-D routes carrying the inter-area P2MP segmented next-hop community.

SEE ALSO

- Example: Configuring Segmented Inter-Area P2MP LSP | **634**
- Configuring Segmented Inter-Area P2MP LSP | **631**
- *all-regions*
- *inter-region*
- *inter-region-segmented*
- *inter-region-template*
- *region*
- *template*

Configuring Segmented Inter-Area P2MP LSP

In order to connect PE routers that are in different areas but in the same AS and that require P2MP connectivity, Junos OS allows you to segment the P2MP LSPs at the area boundary as described in Internet draft *draft-ietf-mpls-seamless-mcast-14.txt* .

To configure segmented inter-area P2MP LSPs at the ingress area segment, the backbone area segment, and the egress area segment, you must do the following:

1. Configure inter-region-segmented for group, wildcard-group-inet, or wildcard-group-inet6 of selective tunnel.
    - Configure inter-region-segmented fan-out and threshold values for a multicast source or wildcard source belonging to group.
        - Specify fan-out and threshold values for a multicast source.

```
[edit routing-instances instance-name provider-tunnel selective]
user@host# set group multicast IP address source source IP address inter-region-
segmented fan-out fan-out value
user@host# set group multicast IP address source source IP address inter-region-
segmented threshold rate-value
```

  - Specify fan-out and threshold values for a wildcard source.

```
[edit routing-instances instance-name provider-tunnel selective]
user@host# set group multicast IP address source wildcard-source inter-region-
segmented fan-out fan-out value
user@host# set group multicast IP address source wildcard-source inter-region-
segmented threshold rate-value
```

- Configure inter-region-segmented fan-out value for wildcard-group-inet belonging to group.

```
[edit routing-instances instance-name provider-tunnel selective]
user@host# set wildcard-group-inet wildcard-source inter-region-segmented fan-out fan-out
value
```

- Configure inter-region-segmented fan-out value for wildcard-group-inet6 belonging to group.

```
[edit routing-instances instance-name provider-tunnel selective]
user@host# set wildcard-group-inet6 wildcard-source inter-region-segmented fan-out fan-out
value
```

2. Configure the inter-region template on the transit ABR to specify the tunnel type to be used for a specific region or for all regions.

   - Configure the inter-region template to specify the tunnel type such as ingress-replication, ldp-p2mp, and rsvp-te for a specific region.

     - Specify create-new-ucast-tunnel or label-switched-path for tunnel type ingress-replication for a specific region.

```
[edit protocols mvpn]
user@host# set inter-region-template template template-name region region-name ingress-
replication create-new-ucast-tunnel
user@host# set inter-region-template template template-name region region-name ingress-
replication label-switched-path label-switched-path-template ( default-template |
default-template)
```

  - Specify tunnel type ldp-p2mp for a specific region.

    ```
    [edit protocols mvpn]
    user@host# set inter-region-template template template-name region region-name ldp-p2mp
    ```

  - Specify static lsp or template for label-switched-path-template for tunnel type rsvp-te belonging to a specific region.

    ```
    [edit protocols mvpn]
    user@host# set inter-region-template template template-name region region-name rsvp-te
    label-switched-path-template (default | lsp-template-name)
    user@host# set inter-region-template template template-name region region-name rsvp-te
    static-lsp static-lsp
    ```

- Configure the inter-region template to specify the tunnel type such as ingress-replication, ldp-p2mp, and rsvp-te for all regions.

  - Specify create-new-ucast-tunnel or label-switched-path for tunnel type ingress-replication for all regions.

    ```
    [edit protocols mvpn]
    user@host# set inter-region-template template template-name all-regions region-name
    ingress-replication create-new-ucast-tunnel
    user@host# set inter-region-template template template-name all-regions region-name
    ingress-replication label-switched-path label-switched-path-template ( default-
    template | default-template)
    ```

  - Specify tunnel type ldp-p2mp for all regions.

    ```
    [edit protocols mvpn]
    user@host# set inter-region-template template template-name all-regions region-name
    ldp-p2mp
    ```

  - Specify static lsp or template for label-switched-path-template for tunnel type rsvp-te belonging to all regions.

    ```
    [edit protocols mvpn]
    user@host# set inter-region-template template template-name all-regions region-name
    ```

```
rsvp-te label-switched-path-template (default | lsp-template-name)
user@host# set inter-region-template template template-name all-regions region-name
rsvp-te static-lsp static-lsp
```

3. Specify the template, which indicates the tunnel types, to be used in inter-region segmentation on the Transit ABRs.

```
[edit routing-instances instance-name provider-tunnel]
user@host# set inter-region template template-name
```

4. Specify no inter-region segmentation if you do not want the ABR to participate in the inter-region segmentation.

```
[edit routing-instances instance-name provider-tunnel]
user@host# set inter-region no-inter-region-segmentation
```

SEE ALSO

Segmented Inter-Area Point-to-Multipoint Label-Switched Paths Overview | **629**

Example: Configuring Segmented Inter-Area P2MP LSP | **634**

all-regions

inter-region

inter-region-template

inter-region-segmented

region

template

Example: Configuring Segmented Inter-Area P2MP LSP

IN THIS SECTION

This example shows how to segment the P2MP LSPs at the area boundary as described in Internet draft *draft-ietf-mpls-seamless-mcast-14.txt*. You can configure policies on the segmented next-hop extended community (S-NH EC) so that S-PMSI A-D routes with the S-NH EC is reflected by the ABR while all other routes are reflected by other route reflectors.

Requirements

This example uses the following hardware and software components:

- Fourteen MX Series 5G Universal Routing Platforms
- Junos OS Release 15.1 or later running on all the routers

Before you begin:

1. Configure the device interfaces.
2. Configure OSPF.

Overview

IN THIS SECTION

- Topology | **636**

Starting with Junos OS Release 15.1, P2MP LSPs can be segmented at the area boundary. A segmented P2MP LSP consists of ingress area segment(Ingress PE router or ASBR), backbone area segment (Transit ABR), and egress area segment (Egress PE routers or ASBRs). Each of the intra-area segments can be carried over provider tunnels such as P2MP RSVP-TE LSP, P2MP mLDP LSP, or ingress replication. Segmentation of inter-area P2MP LSP occurs when the S-PMSI autodiscovery (AD) routes are advertised, which triggers the inclusion of a new BGP extended community or inter-area P2MP segmented next-hop extended community in the ingress PE router or ASBR, transit ABR, and egress PE routers or ASBRs.

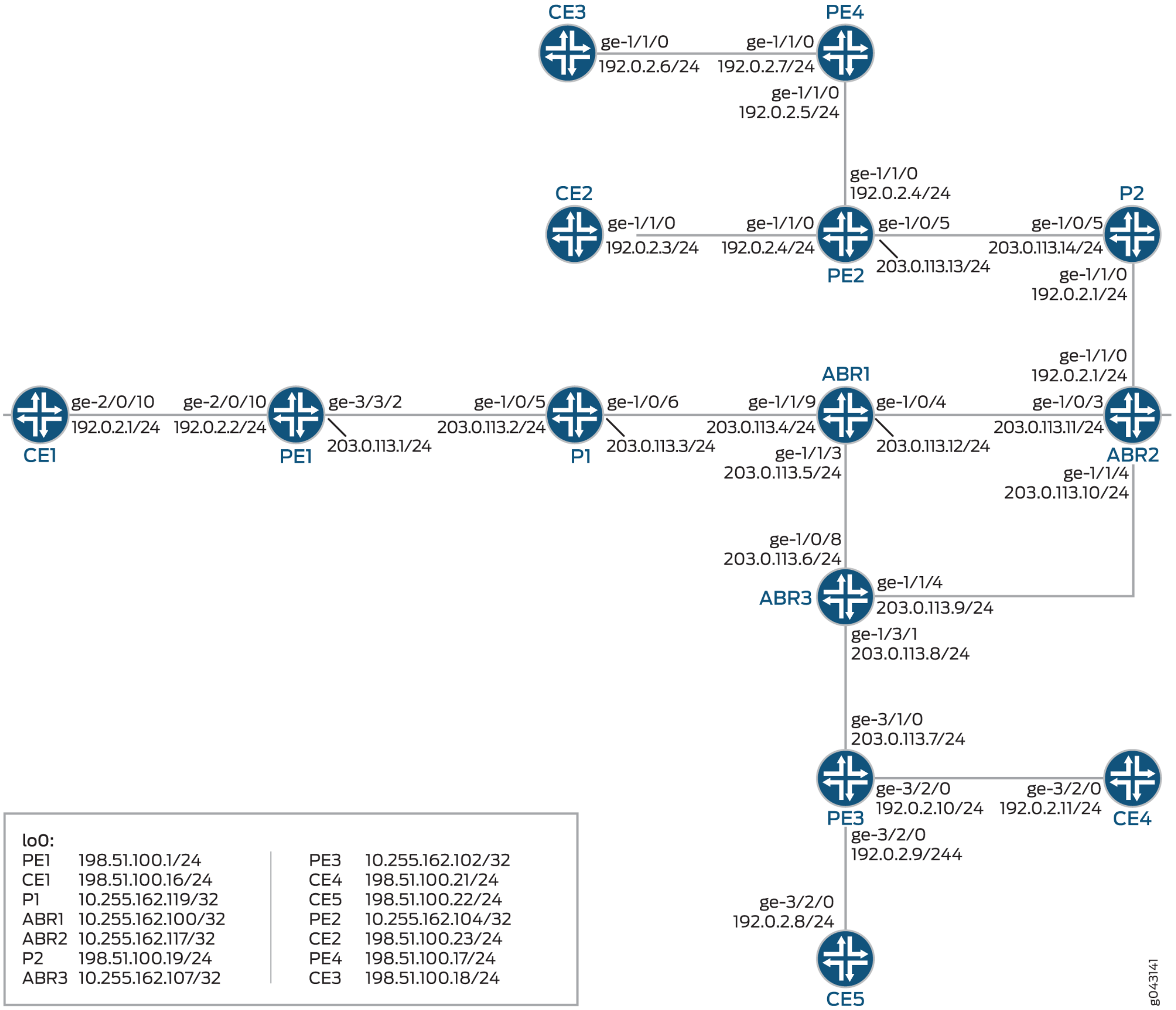
To configure inter-region segmentation at the ingress PE router, configure the `inter-region-segmented` statement at the `[edit routing-instances` *`instance-name`* `provider-tunnel]` hierarchy level. To configure the inter-region template at the transit ABRs, configure the `inter-region-template` *`template-name`* statement at the `[edit protocols mvpn]` hierarchy level. To configure inter-region segmentation at the transit ABR, configure the `inter-region` statement at the `[edit routing-instance` *`instance-name`* `provider-tunnel]` hierarchy level.

Topology

In the topology shown in Figure 56 on page 636, the segmented tunnel combination is as follows:

- Ingress area tunnel — PE1 to ABR1 with IR as the tunnel.
- Backbone area tunnel — ABR1, ABR2, and ABR3 with RSVP-TE as the tunnel.
- Egress area tunnel — ABR2 to PE2 and PE4, ABR3 to PE3 with RSVP-TE as the tunnel.

Figure 56: Example Segmented Inter-Area P2MP LSP



Configuration

IN THIS SECTION

CLI Quick Configuration

To quickly configure this example, copy the following commands, paste them into a text file, remove any line breaks, change any details necessary to match your network configuration, copy and paste the commands into the CLI at the `[edit]` hierarchy level, and then enter `commit` from configuration mode.

PE1

```
set interfaces ge-2/0/10 unit 1 family inet address 192.0.2.2/24
set interfaces ge-2/0/10 unit 1 family inet6 address ::192.0.2.2/120
set interfaces ge-2/0/10 unit 1 family mpls
set interfaces ge-3/3/2 unit 0 family inet address 203.0.113.1/24
set interfaces ge-3/3/2 unit 0 family iso
set interfaces ge-3/3/2 unit 0 family inet6 address ::203.0.113.1/120
set interfaces ge-3/3/2 unit 0 family mpls
set interfaces lo0 unit 201 family inet address 198.51.100.1/24
set routing-options autonomous-system 65550
set protocols rsvp interface fxp0.0 disable
set protocols rsvp interface ge-3/3/2.0
set protocols rsvp interface lo0.0
set protocols mpls ipv6-tunneling
set protocols mpls interface fxp0.0 disable
set protocols mpls interface ge-3/3/2.0
set protocols mpls interface lo0.0
set protocols bgp group IBGP type internal
set protocols bgp group IBGP local-address 10.255.162.109
set protocols bgp group IBGP family inet any
set protocols bgp group IBGP family inet-vpn unicast
set protocols bgp group IBGP family inet-vpn multicast
set protocols bgp group IBGP family inet6 any
set protocols bgp group IBGP family inet6-vpn unicast
```

```
set protocols bgp group IBGP family inet-mvpn signaling
set protocols bgp group IBGP family inet6-mvpn signaling
set protocols bgp group IBGP family inet-mdt signaling
set protocols bgp group IBGP neighbor 10.255.162.100
set protocols ospf traffic-engineering
set protocols ospf area 0.0.0.1 interface fxp0.0 disable
set protocols ospf area 0.0.0.1 interface ge-3/3/2.0
set protocols ospf area 0.0.0.1 interface lo0.0
set protocols ldp interface all
set protocols ldp p2mp
set protocols pim interface all
set protocols pim interface fxp0.0 disable
set protocols pim interface lo0.0
set protocols pim default-vpn-source interface-name lo0.0
set policy-options policy-statement bgp-to-ospf from protocol bgp
set policy-options policy-statement bgp-to-ospf then accept
set routing-instances vpn1 instance-type vrf
set routing-instances vpn1 interface ge-2/0/10
set routing-instances vpn1 interface lo0.201
set routing-instances vpn1 route-distinguisher 10.255.162.109:100
set routing-instances vpn1 provider-tunnel selective group 192.0.2.2/24 source 172.16.1.2/32
ingress-replication label-switched-path
set routing-instances vpn1 provider-tunnel selective group 192.0.2.2/24 source 172.16.1.2/32
threshold-rate 10
set routing-instances vpn1 provider-tunnel selective group 192.0.2.2/24 source 172.16.1.2/32
inter-region-segmented threshold 0
set routing-instances vpn1 provider-tunnel selective group 192.0.2.1/24 source 172.16.1.2/32
ingress-replication label-switched-path
set routing-instances vpn1 provider-tunnel selective group 192.0.2.1/24 source 172.16.1.2/32
threshold-rate 0
set routing-instances vpn1 provider-tunnel selective group 192.0.2.1/24 source 172.16.1.2/32
inter-region-segmented threshold 10
set routing-instances vpn1 provider-tunnel selective group 192.0.2.3/24 source 172.16.1.2/32
ingress-replication label-switched-path
set routing-instances vpn1 provider-tunnel selective group 192.0.2.3/24 source 172.16.1.2/32
threshold-rate 0
set routing-instances vpn1 provider-tunnel selective group 192.0.2.3/24 source 172.16.1.2/32
inter-region-segmented threshold 0
set routing-instances vpn1 provider-tunnel family inet ingress-replication label-switched-path
set routing-instances vpn1 provider-tunnel family inet6 ingress-replication label-switched-path
set routing-instances vpn1 vrf-target target:123:1
set routing-instances vpn1 vrf-table-label
set routing-instances vpn1 protocols ospf export bgp-to-ospf
```

```
set routing-instances vpn1 protocols ospf area 0.0.0.1 interface all
set routing-instances vpn1 protocols ospf area 0.0.0.1 interface lo0.201
set routing-instances vpn1 protocols ospf3 export bgp-to-ospf
set routing-instances vpn1 protocols ospf3 area 0.0.0.1 interface all
set routing-instances vpn1 protocols pim dense-groups 192.0.2.39/24
set routing-instances vpn1 protocols pim dense-groups 192.0.2.40/24
set routing-instances vpn1 protocols pim rp local family inet address 198.51.100.1
set routing-instances vpn1 protocols pim rp static address ::198.51.100.1
set routing-instances vpn1 protocols pim interface all mode sparse-dense
```

CE1

```
set interfaces ge-2/0/3 unit 0 family inet address 172.16.1.1/24
set interfaces ge-2/0/3 unit 0 family iso
set interfaces ge-2/0/3 unit 0 family inet6 address 0000:0000:0000:0000:172:2:1:1/120
set interfaces ge-2/0/3 unit 0 family mpls
set interfaces ge-2/0/10 unit 101 family inet address 192.0.2.1/24
set interfaces ge-2/0/10 unit 101 family inet6 address ::192.0.2.1/120
set interfaces lo0 unit 1 family inet address 198.51.100.16/24
set interfaces lo0 unit 1 family inet6 address abcd::198:51:100:16/128
set protocols igmp interface ge-2/0/3.0 version 3
set protocols ospf area 0.0.0.1 interface all
set protocols ospf3 area 0.0.0.1 interface all
set protocols pim dense-groups 192.0.2.39/24
set protocols pim dense-groups 192.0.2.40/24
set protocols pim rp static address 198.51.100.1
set protocols pim rp static address ::198.51.100.1
set protocols pim interface all mode sparse-dense
set protocols pim interface ge-2/0/10.101
set protocols pim interface ge-2/0/3.0
set protocols pim interface lo0.1
```

P1

```
set interfaces ge-1/0/5 unit 0 family inet address 203.0.113.2/24
set interfaces ge-1/0/5 unit 0 family iso
set interfaces ge-1/0/5 unit 0 family inet6 address ::203.0.113.2/120
set interfaces ge-1/0/5 unit 0 family mpls
set interfaces ge-1/0/6 unit 0 family inet address 203.0.113.3/24
set interfaces ge-1/0/6 unit 0 family iso
set interfaces ge-1/0/6 unit 0 family inet6 address ::203.0.113.3/120
```

```
set interfaces ge-1/0/6 unit 0 family mpls
set interfaces lo0 unit 0 family inet address 203.0.113.0/24
set interfaces lo0 unit 0 family inet address 10.255.162.119/32 primary
set protocols rsvp interface fxp0.0 disable
set protocols rsvp interface ge-1/0/5.0
set protocols rsvp interface ge-1/0/6.0
set protocols rsvp interface lo0.0
set protocols mpls ipv6-tunneling
set protocols mpls interface fxp0.0 disable
set protocols mpls interface ge-1/0/5.0
set protocols mpls interface ge-1/0/6.0
set protocols mpls interface lo0.0
set protocols ospf traffic-engineering
set protocols ospf area 0.0.0.1 interface fxp0.0 disable
set protocols ospf area 0.0.0.1 interface ge-1/0/5.0
set protocols ospf area 0.0.0.1 interface ge-1/0/6.0
set protocols ospf area 0.0.0.1 interface lo0.0
set protocols ldp interface all
set protocols ldp p2mp
set protocols pim interface all
set protocols pim interface fxp0.0 disable
set protocols pim interface lo0.0
```

ABR1

```
set interfaces ge-1/0/4 unit 0 family inet address 203.0.113.12/24
set interfaces ge-1/0/4 unit 0 family iso
set interfaces ge-1/0/4 unit 0 family inet6 address ::203.0.113.12/120
set interfaces ge-1/0/4 unit 0 family mpls
set interfaces ge-1/1/3 unit 0 family inet address 203.0.113.5/24
set interfaces ge-1/1/3 unit 0 family iso
set interfaces ge-1/1/3 unit 0 family inet6 address ::203.0.113.5/120
set interfaces ge-1/1/3 unit 0 family mpls
set interfaces ge-1/1/9 unit 0 family inet address 203.0.113.4/24
set interfaces ge-1/1/9 unit 0 family iso
set interfaces ge-1/1/9 unit 0 family inet6 address ::203.0.113.4/120
set interfaces ge-1/1/9 unit 0 family mpls
set interfaces lo0 unit 0 family inet address 203.0.113.0/24
set interfaces lo0 unit 0 family inet address 10.255.162.100/32 primary
set routing-options autonomous-system 65550
set protocols rsvp interface fxp0.0 disable
set protocols rsvp interface ge-1/1/9.0
```

```
set protocols rsvp interface ge-1/0/4.0
set protocols rsvp interface ge-1/1/3.0
set protocols rsvp interface lo0.0
set protocols rsvp interface all
set protocols mpls ipv6-tunneling
set protocols mpls interface fxp0.0 disable
set protocols mpls interface ge-1/1/9.0
set protocols mpls interface ge-1/0/4.0
set protocols mpls interface ge-1/1/3.0
set protocols mpls interface lo0.0
set protocols mpls interface all
set protocols bgp group IBGP_1 type internal
set protocols bgp group IBGP_1 local-address 10.255.162.100
set protocols bgp group IBGP_1 family inet any
set protocols bgp group IBGP_1 family inet-vpn unicast
set protocols bgp group IBGP_1 family inet-vpn multicast
set protocols bgp group IBGP_1 family inet6 any
set protocols bgp group IBGP_1 family inet6-vpn unicast
set protocols bgp group IBGP_1 family inet-mvpn signaling
set protocols bgp group IBGP_1 family inet6-mvpn signaling
set protocols bgp group IBGP_1 family inet-mdt signaling
set protocols bgp group IBGP_1 cluster 0.0.0.1
set protocols bgp group IBGP_1 neighbor 10.255.162.109
set protocols bgp group IBGP_0 type internal
set protocols bgp group IBGP_0 local-address 10.255.162.100
set protocols bgp group IBGP_0 family inet any
set protocols bgp group IBGP_0 family inet-vpn unicast
set protocols bgp group IBGP_0 family inet-vpn multicast
set protocols bgp group IBGP_0 family inet6 any
set protocols bgp group IBGP_0 family inet6-vpn unicast
set protocols bgp group IBGP_0 family inet-mvpn signaling
set protocols bgp group IBGP_0 family inet6-mvpn signaling
set protocols bgp group IBGP_0 family inet-mdt signaling
set protocols bgp group IBGP_0 neighbor 10.255.162.117
set protocols bgp group IBGP_0 neighbor 10.255.162.107
set protocols ospf traffic-engineering
set protocols ospf area 0.0.0.1 interface fxp0.0 disable
set protocols ospf area 0.0.0.1 interface ge-1/1/9.0
set protocols ospf area 0.0.0.0 interface ge-1/0/4.0
set protocols ospf area 0.0.0.0 interface ge-1/1/3.0
set protocols ospf area 0.0.0.0 interface lo0.0
set protocols ldp interface all
set protocols ldp p2mp
```

```
set protocols pim interface all
set protocols pim interface fxp0.0 disable
set protocols pim interface lo0.0
set protocols mvpn inter-region-template template template_1 region IBGP_0 rsvp-te label-
switched-path-template default-template
set protocols mvpn inter-region-template template template_2 region IBGP_0 ldp-p2mp
set protocols mvpn inter-region-template template template_3 region IBGP_0 ingress-replication
create-new-ucast-tunnel
set protocols mvpn inter-region-template template template_3 region IBGP_0 ingress-replication
label-switched-path label-switched-path-template default-template
set protocols mvpn inter-region-template template template_4 all-regions incoming
set protocols mvpn inter-region-template template template_5 region IBGP_0 rsvp-te static-lsp
ABR1_to_ABR3
set routing-instances vpn1 instance-type vrf
set routing-instances vpn1 route-distinguisher 10.255.162.100:100
set routing-instances vpn1 provider-tunnel inter-region template template_1
set routing-instances vpn1 vrf-target target:123:1
set routing-instances vpn1 vrf-table-label
```

ABR2

```

set interfaces ge-1/0/3 unit 0 family inet address 203.0.113.11/24
set interfaces ge-1/0/3 unit 0 family iso
set interfaces ge-1/0/3 unit 0 family inet6 address ::203.0.113.11/120
set interfaces ge-1/0/3 unit 0 family mpls
set interfaces ge-1/1/4 unit 0 family inet address 203.0.113.10/24
set interfaces ge-1/1/4 unit 0 family iso
set interfaces ge-1/1/4 unit 0 family inet6 address ::203.0.113.10/120
set interfaces ge-1/1/4 unit 0 family mpls
set interfaces ge-1/1/10 unit 1 family inet address 192.0.2.2/24
set interfaces ge-1/1/10 unit 1 family inet6 address ::192.0.2.2/120
set interfaces ge-1/1/10 unit 1 family mpls
set interfaces lo0 unit 0 family inet address 203.0.113.0/24
set interfaces lo0 unit 0 family inet address 10.255.162.117/32 primary
set routing-options autonomous-system 65550
set protocols rsvp interface fxp0.0 disable
set protocols rsvp interface lo0.0
set protocols rsvp interface all
set protocols mpls ipv6-tunneling
set protocols mpls interface fxp0.0 disable
set protocols mpls interface lo0.0
set protocols mpls interface all
```

```
set protocols bgp group IBGP_2 type internal
set protocols bgp group IBGP_2 local-address 10.255.162.117
set protocols bgp group IBGP_2 family inet any
set protocols bgp group IBGP_2 family inet-vpn unicast
set protocols bgp group IBGP_2 family inet-vpn multicast
set protocols bgp group IBGP_2 family inet6 any
set protocols bgp group IBGP_2 family inet6-vpn unicast
set protocols bgp group IBGP_2 family inet-mvpn signaling
set protocols bgp group IBGP_2 family inet6-mvpn signaling
set protocols bgp group IBGP_2 family inet-mdt signaling
set protocols bgp group IBGP_2 cluster 0.0.0.2
set protocols bgp group IBGP_2 neighbor 10.255.162.104
set protocols bgp group IBGP_2 neighbor 198.51.100.17
set protocols bgp group IBGP_0 type internal
set protocols bgp group IBGP_0 local-address 10.255.162.117
set protocols bgp group IBGP_0 family inet any
set protocols bgp group IBGP_0 family inet-vpn unicast
set protocols bgp group IBGP_0 family inet-vpn multicast
set protocols bgp group IBGP_0 family inet6 any
set protocols bgp group IBGP_0 family inet6-vpn unicast
set protocols bgp group IBGP_0 family inet-mvpn signaling
set protocols bgp group IBGP_0 family inet6-mvpn signaling
set protocols bgp group IBGP_0 family inet-mdt signaling
set protocols bgp group IBGP_0 neighbor 10.255.162.100
set protocols bgp group IBGP_0 neighbor 10.255.162.107
set protocols ospf traffic-engineering
set protocols ospf area 0.0.0.0 interface fxp0.0 disable
set protocols ospf area 0.0.0.0 interface ge-1/0/3.0
set protocols ospf area 0.0.0.0 interface ge-1/1/4.0
set protocols ospf area 0.0.0.0 interface lo0.0
set protocols ospf area 0.0.0.2 interface ge-1/1/10.1
set protocols ldp interface all
set protocols ldp p2mp
set protocols pim interface all
set protocols pim interface fxp0.0 disable
set protocols pim interface lo0.0
set protocols mvpn inter-region-template template template_1 region IBGP_2 rsvp-te label-
switched-path-template default-template
set protocols mvpn inter-region-template template template_2 region IBGP_2 ldp-p2mp
set protocols mvpn inter-region-template template template_3 region IBGP_2 ingress-replication
create-new-ucast-tunnel
set protocols mvpn inter-region-template template template_3 region IBGP_2 ingress-replication
label-switched-path label-switched-path-template default-template
```

```
set protocols mvpn inter-region-template template template_4 all-regions incoming
set protocols mvpn inter-region-template template template_5 region IBGP_2 rsvp-te static-lsp
ABR2_to_PE2_3
set routing-instances vpn1 instance-type vrf
set routing-instances vpn1 route-distinguisher 10.255.162.117:100
set routing-instances vpn1 provider-tunnel inter-region template template_1
set routing-instances vpn1 vrf-target target:123:1
set routing-instances vpn1 vrf-table-label
```

P2

```
set interfaces ge-1/0/5 unit 0 family inet address 203.0.113.14/24
set interfaces ge-1/0/5 unit 0 family iso
set interfaces ge-1/0/5 unit 0 family inet6 address ::203.0.113.14/120
set interfaces ge-1/0/5 unit 0 family mpls
set interfaces ge-1/1/10 unit 101 family inet address 192.0.2.1/24
set interfaces ge-1/1/10 unit 101 family inet6 address ::192.0.2.1/120
set interfaces ge-1/1/10 unit 101 family mpls
set interfaces lo0 unit 1 family inet address 198.51.100.19/24
set interfaces lo0 unit 1 family inet6 address abcd::198:51:100:19/128
set protocols rsvp interface fxp0.0 disable
set protocols rsvp interface all
set protocols rsvp interface lo0.1
set protocols mpls ipv6-tunneling
set protocols mpls interface fxp0.0 disable
set protocols mpls interface all
set protocols mpls interface lo0.1
set protocols ospf traffic-engineering
set protocols ospf area 0.0.0.2 interface all
set protocols ospf area 0.0.0.2 interface lo0.1
set protocols ldp interface all
set protocols ldp p2mp
set protocols pim dense-groups 192.0.2.39/24
set protocols pim dense-groups 192.0.2.40/24
set protocols pim rp static address 198.51.100.1
set protocols pim rp static address ::198.51.100.1
set protocols pim interface all mode sparse-dense
```

ABR3

```
set interfaces ge-1/0/8 unit 0 family inet address 203.0.113.6/24
set interfaces ge-1/0/8 unit 0 family iso
set interfaces ge-1/0/8 unit 0 family inet6 address ::203.0.113.6/120
set interfaces ge-1/0/8 unit 0 family mpls
set interfaces ge-1/1/4 unit 0 family inet address 203.0.113.9/24
set interfaces ge-1/1/4 unit 0 family iso
set interfaces ge-1/1/4 unit 0 family inet6 address ::203.0.113.9/120
set interfaces ge-1/1/4 unit 0 family mpls
set interfaces ge-1/3/1 unit 0 family inet address 203.0.113.8/24
set interfaces ge-1/3/1 unit 0 family iso
set interfaces ge-1/3/1 unit 0 family inet6 address ::203.0.113.8/120
set interfaces ge-1/3/1 unit 0 family mpls
set interfaces lo0 unit 0 family inet address 203.0.113.0/24
set interfaces lo0 unit 0 family inet address 10.255.162.107/32 primary
set routing-options autonomous-system 65550
set protocols rsvp interface all
set protocols rsvp interface fxp0.0 disable
set protocols rsvp interface lo0.0
set protocols mpls ipv6-tunneling
set protocols mpls label-switched-path ABR3_to_PE3 from 10.255.162.107
set protocols mpls label-switched-path ABR3_to_PE3 to 10.255.162.102
set protocols mpls label-switched-path ABR3_to_PE3 p2mp vpn1
set protocols mpls label-switched-path ABR3_to_ABR1 from 10.255.162.107
set protocols mpls label-switched-path ABR3_to_ABR1 to 10.255.162.100
set protocols mpls label-switched-path ABR3_to_ABR1 p2mp vpn1
set protocols mpls label-switched-path ABR3_to_ABR2 from 10.255.162.107
set protocols mpls label-switched-path ABR3_to_ABR2 to 10.255.162.117
set protocols mpls label-switched-path ABR3_to_ABR2 p2mp vpn1
set protocols mpls interface all
set protocols mpls interface fxp0.0 disable
set protocols mpls interface lo0.0
set protocols bgp group IBGP_3 type internal
set protocols bgp group IBGP_3 local-address 10.255.162.107
set protocols bgp group IBGP_3 family inet any
set protocols bgp group IBGP_3 family inet-vpn unicast
set protocols bgp group IBGP_3 family inet-vpn multicast
set protocols bgp group IBGP_3 family inet6 any
set protocols bgp group IBGP_3 family inet6-vpn unicast
set protocols bgp group IBGP_3 family inet-mvpn signaling
set protocols bgp group IBGP_3 family inet6-mvpn signaling
```

```
set protocols bgp group IBGP_3 family inet-mdt signaling
set protocols bgp group IBGP_3 cluster 0.0.0.3
set protocols bgp group IBGP_3 neighbor 10.255.162.102
set protocols bgp group IBGP_0 type internal
set protocols bgp group IBGP_0 local-address 10.255.162.107
set protocols bgp group IBGP_0 family inet any
set protocols bgp group IBGP_0 family inet-vpn unicast
set protocols bgp group IBGP_0 family inet-vpn multicast
set protocols bgp group IBGP_0 family inet6 any
set protocols bgp group IBGP_0 family inet6-vpn unicast
set protocols bgp group IBGP_0 family inet-mvpn signaling
set protocols bgp group IBGP_0 family inet6-mvpn signaling
set protocols bgp group IBGP_0 family inet-mdt signaling
set protocols bgp group IBGP_0 neighbor 10.255.162.100
set protocols bgp group IBGP_0 neighbor 10.255.162.117
set protocols ospf traffic-engineering
set protocols ospf area 0.0.0.0 interface fxp0.0 disable
set protocols ospf area 0.0.0.0 interface ge-1/0/8.0
set protocols ospf area 0.0.0.0 interface ge-1/1/4.0
set protocols ospf area 0.0.0.0 interface lo0.0
set protocols ospf area 0.0.0.3 interface ge-1/3/1.0
set protocols ldp interface all
set protocols ldp p2mp
set protocols pim interface all
set protocols pim interface fxp0.0 disable
set protocols pim interface lo0.0
set protocols mvpn inter-region-template template template_1 region IBGP_3 rsvp-te label-
switched-path-template default-template
set protocols mvpn inter-region-template template template_2 region IBGP_3 ldp-p2mp
set protocols mvpn inter-region-template template template_3 region IBGP_3 ingress-replication
create-new-ucast-tunnel
set protocols mvpn inter-region-template template template_3 region IBGP_3 ingress-replication
label-switched-path label-switched-path-template default-template
set protocols mvpn inter-region-template template template_4 all-regions incoming
set protocols mvpn inter-region-template template template_5 region IBGP_3 rsvp-te static-lsp
ABR3_to_PE3
set routing-instances vpn1 instance-type vrf
set routing-instances vpn1 route-distinguisher 10.255.162.107:100
set routing-instances vpn1 provider-tunnel inter-region template template_1
set routing-instances vpn1 vrf-target target:123:1
set routing-instances vpn1 vrf-table-label
```

PE3

```
set interfaces ge-3/0/1 unit 0 family inet address 203.0.113.15/24
set interfaces ge-3/0/1 unit 0 family iso
set interfaces ge-3/0/1 unit 0 family inet6 address ::203.0.113.15/120
set interfaces ge-3/0/1 unit 0 family mpls
set interfaces ge-3/1/0 unit 0 family inet address 203.0.113.7/24
set interfaces ge-3/1/0 unit 0 family iso
set interfaces ge-3/1/0 unit 0 family inet6 address ::203.0.113.7/120
set interfaces ge-3/1/0 unit 0 family mpls
set interfaces ge-3/2/0 unit 1 family inet address 192.0.2.9/24
set interfaces ge-3/2/0 unit 1 family inet6 address ::192.0.2.9/120
set interfaces ge-3/2/0 unit 1 family mpls
set interfaces ge-3/2/0 unit 2 family inet address 192.0.2.10/24
set interfaces ge-3/2/0 unit 2 family inet6 address ::192.0.2.10/120
set interfaces ge-3/2/0 unit 2 family mpls
set interfaces lo0 unit 0 family inet address 203.0.113.0/24
set interfaces lo0 unit 0 family inet address 10.255.162.102/32 primary
set routing-options autonomous-system 65550
set protocols rsvp interface fxp0.0 disable
set protocols rsvp interface all
set protocols rsvp interface lo0.0
set protocols mpls ipv6-tunneling
set protocols mpls label-switched-path PE3_to_PE2 from 10.255.162.102
set protocols mpls label-switched-path PE3_to_PE2 to 10.255.162.104
set protocols mpls label-switched-path PE3_to_PE2 p2mp vpn1
set protocols mpls label-switched-path PE3_to_PE4 from 10.255.162.102
set protocols mpls label-switched-path PE3_to_PE4 to 198.51.100.17
set protocols mpls label-switched-path PE3_to_PE4 p2mp vpn1
set protocols mpls label-switched-path PE3_to_PE1 from 10.255.162.102
set protocols mpls label-switched-path PE3_to_PE1 to 10.255.162.109
set protocols mpls label-switched-path PE3_to_PE1 p2mp vpn1
set protocols mpls label-switched-path PE3_to_ABR3 from 10.255.162.102
set protocols mpls label-switched-path PE3_to_ABR3 to 10.255.162.107
set protocols mpls label-switched-path PE3_to_ABR3 p2mp vpn1
set protocols mpls interface fxp0.0 disable
set protocols mpls interface all
set protocols mpls interface lo0.0
set protocols bgp group IBGP type internal
set protocols bgp group IBGP local-address 10.255.162.102
set protocols bgp group IBGP family inet any
set protocols bgp group IBGP family inet-vpn unicast
```

```
set protocols bgp group IBGP family inet-vpn multicast
set protocols bgp group IBGP family inet6 any
set protocols bgp group IBGP family inet6-vpn unicast
set protocols bgp group IBGP family inet-mvpn signaling
set protocols bgp group IBGP family inet6-mvpn signaling
set protocols bgp group IBGP family inet-mdt signaling
set protocols bgp group IBGP neighbor 10.255.162.107
set protocols ospf traffic-engineering
set protocols ospf area 0.0.0.3 interface fxp0.0 disable
set protocols ospf area 0.0.0.3 interface all
set protocols ospf area 0.0.0.3 interface lo0.0
set protocols ldp interface all
set protocols ldp p2mp
set protocols pim interface all
set protocols pim interface fxp0.0 disable
set protocols pim interface lo0.0
set protocols pim default-vpn-source interface-name lo0.0
set policy-options policy-statement bgp-to-ospf from protocol bgp
set policy-options policy-statement bgp-to-ospf then accept
set routing-instances vpn1 instance-type vrf
set routing-instances vpn1 interface ge-3/2/0.1
set routing-instances vpn1 interface ge-3/2/0.2
set routing-instances vpn1 route-distinguisher 10.255.162.102:100
set routing-instances vpn1 provider-tunnel family inet ingress-replication label-switched-path
set routing-instances vpn1 provider-tunnel family inet6 ingress-replication label-switched-path
set routing-instances vpn1 vrf-target target:123:1
set routing-instances vpn1 vrf-table-label
set routing-instances vpn1 protocols ospf export bgp-to-ospf
set routing-instances vpn1 protocols ospf area 0.0.0.3 interface all
set routing-instances vpn1 protocols ospf3 export bgp-to-ospf
set routing-instances vpn1 protocols ospf3 area 0.0.0.3 interface all
set routing-instances vpn1 protocols pim dense-groups 192.0.2.39/24
set routing-instances vpn1 protocols pim dense-groups 192.0.2.40/24
set routing-instances vpn1 protocols pim rp static address 198.51.100.1
set routing-instances vpn1 protocols pim rp static address ::198.51.100.1
set routing-instances vpn1 protocols pim interface all mode sparse-dense
set routing-instances vpn1 protocols mvpn mvpn-mode spt-only
```

CE4

```
set interfaces ge-3/1/1 unit 0 family inet address 172.16.0.1/24
set interfaces ge-3/1/1 unit 0 family iso
```

```
set interfaces ge-3/1/1 unit 0 family inet6 address 0000:0000:0000:0000:172:16:0:1/120
set interfaces ge-3/1/1 unit 0 family mpls
set interfaces ge-3/2/0 unit 102 description "Link to PE3_1 from CE3_2"
set interfaces ge-3/2/0 unit 102 family inet address 192.0.2.11/24
set interfaces ge-3/2/0 unit 102 family inet6 address ::192.0.2.11/120
set interfaces ge-3/2/0 unit 102 family mpls
set interfaces lo0 unit 2 family inet address 198.51.100.21/24
set interfaces lo0 unit 2 family inet6 address abcd::198:51:100:21/128
set protocols igmp interface ge-3/1/1.0 version 3
set protocols mld interface ge-3/1/1.0 version 2
set protocols ospf area 0.0.0.3 interface all
set protocols ospf3 area 0.0.0.3 interface all
set protocols pim dense-groups 192.0.2.39/24
set protocols pim dense-groups 192.0.2.40/24
set protocols pim rp static address 198.51.100.1
set protocols pim rp static address ::198.51.100.1
set protocols pim interface all mode sparse-dense
```

CE5

```
set interfaces ge-3/2/0 unit 101 family inet address 192.0.2.8/24
set interfaces ge-3/2/0 unit 101 family inet6 address ::192.0.2.8/120
set interfaces ge-3/2/0 unit 101 family mpls
set interfaces lo0 unit 1 family inet address 198.51.100.22/24
set interfaces lo0 unit 1 family inet6 address abcd::198:51:100:22/128
set protocols ospf area 0.0.0.3 interface all
set protocols ospf3 area 0.0.0.3 interface all
set protocols pim dense-groups 192.0.2.39/24
set protocols pim dense-groups 192.0.2.40/24
set protocols pim rp static address 198.51.100.1
set protocols pim rp static address ::198.51.100.1
set protocols pim interface all mode sparse-dense
```

PE2

```
set interfaces ge-1/0/5 unit 0 family inet address 203.0.113.13/24
set interfaces ge-1/0/5 unit 0 family iso
set interfaces ge-1/0/5 unit 0 family inet6 address ::203.0.113.13/120
set interfaces ge-1/0/5 unit 0 family mpls
set interfaces ge-1/1/00 unit 1 family inet address 192.0.2.4/24
set interfaces ge-1/1/0 unit 1 family inet6 address ::192.0.2.4/120
```

```
set interfaces ge-1/1/0 unit 1 family mpls
set interfaces ge-1/1/0 unit 2 family inet address 192.0.2.12/24
set interfaces ge-1/1/0 unit 2 family inet6 address ::192.0.2.12/120
set interfaces ge-1/1/0 unit 2 family mpls
set interfaces vt-1/1/0 unit 1 family inet
set interfaces vt-1/1/0 unit 1 family inet6
set interfaces lo0 unit 0 family inet address 203.0.113.0/24
set interfaces lo0 unit 0 family inet address 10.255.162.104/24 primary
set interfaces lo0 unit 201 family inet6 address ::198.51.100.1/128
set routing-options autonomous-system 65550
set protocols rsvp interface fxp0.0 disable
set protocols rsvp interface all
set protocols rsvp interface lo0.0
set protocols mpls ipv6-tunneling
set protocols mpls label-switched-path PE2_to_PE3 from 10.255.162.104
set protocols mpls label-switched-path PE2_to_PE3 to 10.255.162.102
set protocols mpls label-switched-path PE2_to_PE3 p2mp vpn1
set protocols mpls label-switched-path PE2_to_PE4 from 10.255.162.104
set protocols mpls label-switched-path PE2_to_PE4 to 198.51.100.17
set protocols mpls label-switched-path PE2_to_PE4 p2mp vpn1
set protocols mpls label-switched-path PE2_to_PE1 from 10.255.162.104
set protocols mpls label-switched-path PE2_to_PE1 to 10.255.162.109
set protocols mpls label-switched-path PE2_to_PE1 p2mp vpn1
set protocols mpls label-switched-path PE2_to_ABR2 from 10.255.162.104
set protocols mpls label-switched-path PE2_to_ABR2 to 10.255.162.117
set protocols mpls label-switched-path PE2_to_ABR2 p2mp vpn1
set protocols mpls interface fxp0.0 disable
set protocols mpls interface all
set protocols mpls interface lo0.0
set protocols bgp group IBGP type internal
set protocols bgp group IBGP local-address 10.255.162.104
set protocols bgp group IBGP family inet any
set protocols bgp group IBGP family inet-vpn unicast
set protocols bgp group IBGP family inet-vpn multicast
set protocols bgp group IBGP family inet6 any
set protocols bgp group IBGP family inet6-vpn unicast
set protocols bgp group IBGP family inet-mvpn signaling
set protocols bgp group IBGP family inet6-mvpn signaling
set protocols bgp group IBGP family inet-mdt signaling
set protocols bgp group IBGP neighbor 10.255.162.117
set protocols ospf traffic-engineering
set protocols ospf area 0.0.0.2 interface fxp0.0 disable
set protocols ospf area 0.0.0.2 interface all
```

```
set protocols ospf area 0.0.0.2 interface lo0.0
set protocols ldp interface all
set protocols ldp p2mp
set protocols pim interface all
set protocols pim interface fxp0.0 disable
set protocols pim interface lo0.0
set protocols pim default-vpn-source interface-name lo0.0
set policy-options policy-statement bgp-to-ospf from protocol bgp
set policy-options policy-statement bgp-to-ospf then accept
set routing-instances vpn1 instance-type vrf
set routing-instances vpn1 interface ge-1/1/0.1
set routing-instances vpn1 interface vt-1/1/0.1 multicast
set routing-instances vpn1 interface lo0.201
set routing-instances vpn1 route-distinguisher 10.255.162.104:100
set routing-instances vpn1 provider-tunnel family inet ingress-replication label-switched-path
set routing-instances vpn1 provider-tunnel family inet6 ingress-replication label-switched-path
set routing-instances vpn1 vrf-target target:123:1
set routing-instances vpn1 vrf-table-label
set routing-instances vpn1 protocols ospf export bgp-to-ospf
set routing-instances vpn1 protocols ospf area 0.0.0.2 interface all
set routing-instances vpn1 protocols ospf area 0.0.0.2 interface lo0.201
set routing-instances vpn1 protocols ospf3 export bgp-to-ospf
set routing-instances vpn1 protocols ospf3 area 0.0.0.2 interface all
set routing-instances vpn1 protocols pim dense-groups 192.0.2.39/24
set routing-instances vpn1 protocols pim dense-groups 192.0.2.40/24
set routing-instances vpn1 protocols pim rp local family inet6 address ::198.51.100.1
set routing-instances vpn1 protocols pim rp static address 198.51.100.1
set routing-instances vpn1 protocols pim interface all mode sparse-dense
set routing-instances vpn1 protocols mvpn mvpn-mode spt-only
```

CE2

```
set interfaces ge-1/0/0 unit 0 family inet address 172.17.1.1/24
set l interfaces ge-1/0/0 unit 0 family iso
set interfaces ge-1/0/0 unit 0 family inet6 address 0000:0000:0000:0000:172:17:1:1/120
set interfaces ge-1/0/0 unit 0 family mpls
set interfaces ge-1/1/0 unit 101 family inet address 192.0.2.3/24
set interfaces ge-1/1/0 unit 101 family inet6 address ::192.0.2.3/120
set interfaces ge-1/1/0 unit 101 family mpls
set interfaces lo0 unit 1 family inet address 198.51.100.23/24
set interfaces lo0 unit 1 family inet6 address abcd::198:51:100:23/128
set protocols igmp interface ge-1/0/0.0 version 3
```

```
set protocols mld interface ge-1/0/0.0 version 2
set protocols ospf area 0.0.0.2 interface all
set protocols ospf3 area 0.0.0.2 interface all
set protocols pim dense-groups 192.0.2.39/24
set protocols pim dense-groups 192.0.2.40/24
set protocols pim rp static address 198.51.100.1
set protocols pim rp static address ::198.51.100.1
set protocols pim interface all mode sparse-dense
```

PE4

```
set interfaces ge-1/1/0 unit 3 family inet address 192.0.2.7/24
set interfaces ge-1/1/0 unit 3 family inet6 address ::192.0.2.7/120
set interfaces ge-1/1/0 unit 3 family mpls
set interfaces ge-1/1/0 unit 102 family inet address 192.0.2.5/24
set interfaces ge-1/1/0 unit 102 family inet6 address ::192.0.2.5/120
set interfaces ge-1/1/0 unit 102 family mpls
set interfaces vt-1/1/0 unit 0 family inet
set interfaces vt-1/1/0 unit 0 family inet6
set interfaces lo0 unit 2 family inet address 198.51.100.17/24
set interfaces lo0 unit 2 family inet6 address abcd::198:51:100:17/128
set protocols rsvp interface fxp0.0 disable
set protocols rsvp interface all
set protocols rsvp interface lo0.2
set protocols mpls ipv6-tunneling
set protocols mpls label-switched-path PE4_to_PE3 from 198.51.100.17
set protocols mpls label-switched-path PE4_to_PE3 to 10.255.162.102
set protocols mpls label-switched-path PE4_to_PE2 from 198.51.100.17
set protocols mpls label-switched-path PE4_to_PE2 to 10.255.162.104
set protocols mpls label-switched-path PE4_to_PE1 from 198.51.100.17
set protocols mpls label-switched-path PE4_to_PE1 to 10.255.162.109
set protocols mpls label-switched-path PE4_to_ABR2 from 198.51.100.17
set protocols mpls label-switched-path PE4_to_ABR2 to 10.255.162.117
set protocols mpls interface fxp0.0 disable
set protocols mpls interface all
set protocols mpls interface lo0.2
set protocols bgp group IBGP type internal
set protocols bgp group IBGP local-address 198.51.100.17
set protocols bgp group IBGP family inet any
set protocols bgp group IBGP family inet-vpn unicast
set protocols bgp group IBGP family inet-vpn multicast
set protocols bgp group IBGP family inet6 any
```

```
set protocols bgp group IBGP family inet6-vpn unicast
set protocols bgp group IBGP family inet-mvpn signaling
set protocols bgp group IBGP family inet6-mvpn signaling
set protocols bgp group IBGP family inet-mdt signaling
set protocols bgp group IBGP neighbor 10.255.162.117
set protocols ospf traffic-engineering
set protocols ospf area 0.0.0.2 interface fxp0.0 disable
set protocols ospf area 0.0.0.2 interface all
set protocols ospf area 0.0.0.2 interface lo0.2
set protocols ldp interface all
set protocols ldp p2mp
set protocols pim interface all
set protocols pim interface fxp0.0 disable
set protocols pim interface lo0.2
set protocols pim default-vpn-source interface-name lo0.2
set policy-options policy-statement bgp-to-ospf from protocol bgp
set policy-options policy-statement bgp-to-ospf then accept
set routing-instances vpn1 instance-type vrf
set routing-instances vpn1 interface vt-1/1/0.0 multicast
set routing-instances vpn1 interface ge-1/1/0.3
set routing-instances vpn1 route-distinguisher 198.51.100.17:100
set routing-instances vpn1 provider-tunnel family inet ingress-replication label-switched-path
set routing-instances vpn1 provider-tunnel family inet6 ingress-replication label-switched-path
set routing-instances vpn1 vrf-target target:123:1
set routing-instances vpn1 vrf-table-label
set routing-instances vpn1 protocols ospf export bgp-to-ospf
set routing-instances vpn1 protocols ospf area 0.0.0.2 interface all
set routing-instances vpn1 protocols ospf3 export bgp-to-ospf
set routing-instances vpn1 protocols ospf3 area 0.0.0.2 interface all
set routing-instances vpn1 protocols pim dense-groups 192.0.2.39/24
set routing-instances vpn1 protocols pim dense-groups 192.0.2.40/24
set routing-instances vpn1 protocols pim rp static address 198.51.100.1
set routing-instances vpn1 protocols pim rp static address ::198.51.100.1
set routing-instances vpn1 protocols pim interface all mode sparse-dense
set routing-instances vpn1 protocols mvpn mvpn-mode spt-only
set routing-options autonomous-system 65550
```

CE3

```
set interfaces ge-1/1/0 unit 103 family inet address 192.0.2.6/24
set interfaces ge-1/1/0 unit 103 family inet6 address ::192.0.2.6/120
set interfaces ge-1/1/0 unit 103 family mpls
```

```
set interfaces ge-2/1/1 unit 0 family inet address 172.17.2.1/24
set interfaces ge-2/1/1 unit 0 family iso
set interfaces ge-2/1/1 unit 0 family inet6 address 0000:0000:0000:0000:172:17:2:1/120
set interfaces ge-2/1/1 unit 0 family mpls
set interfaces lo0 unit 3 family inet address 198.51.100.18/24
set interfaces lo0 unit 3 family inet6 address abcd::198:51:100:18/128
set protocols igmp interface ge-2/1/1.0 version 3
set protocols mld interface ge-2/1/1.0 version 2
set protocols ospf area 0.0.0.2 interface all
set protocols ospf3 area 0.0.0.2 interface all
set protocols pim dense-groups 192.0.2.39/24
set protocols pim dense-groups 192.0.2.40/24
set protocols pim rp static address 198.51.100.1
set protocols pim rp static address ::198.51.100.1
set protocols pim interface all mode sparse-dense
```

Configuring PE1

Step-by-Step Procedure

The following example requires that you navigate various levels in the configuration hierarchy. For information about navigating the CLI, see *Using the CLI Editor in Configuration Mode* in the CLI User Guide.

To configure Device PE1:

1. Configure the interfaces.

```
[edit interfaces]
user@PE1# set ge-2/0/10 unit 1 family inet address 192.0.2.2/24
user@PE1# set ge-2/0/10 unit 1 family inet6 address ::192.0.2.2/120
user@PE1# set ge-2/0/10 unit 1 family mpls
user@PE1# set ge-3/3/2 unit 0 family inet address 203.0.113.1/24
user@PE1# set ge-3/3/2 unit 0 family iso
user@PE1# set ge-3/3/2 unit 0 family inet6 address ::203.0.113.1/120
user@PE1# set ge-3/3/2 unit 0 family mpls
user@PE1# set lo0 unit 201 family inet address 198.51.100.1/24
```

2. Configure the autonomous system number.

```
[edit routing-options]
user@PE1# set autonomous-system 65550
```

3. Disable RSVP on the management interface and enable RSVP on the interfaces.

```
[edit protocols rsvp]
user@PE1# set interface fxp0.0 disable
user@PE1# set interface ge-3/3/2.0
user@PE1# set interface lo0.0
```

4. Enable IPv6 tunneling.

```
[edit protocols mpls]
user@PE1# set ipv6-tunneling
```

5. Disable MPLS on the management interface and enable MPLS on the interfaces.

```
[edit protocols mpls]
user@PE1# set interface fxp0.0 disable
user@PE1# set interface ge-3/3/2.0
user@PE1# set interface lo0.0
```

6. Configure the BGP protocol.

```
[edit protocols bgp]
user@PE1# set group IBGP type internal
user@PE1# set group IBGP local-address 10.255.162.109
user@PE1# set group IBGP family inet any
user@PE1# set group IBGP family inet-vpn unicast
user@PE1# set group IBGP family inet-vpn multicast
user@PE1# set group IBGP family inet6 any
user@PE1# set group IBGP family inet6-vpn unicast
user@PE1# set group IBGP family inet-mvpn signaling
user@PE1# set group IBGP family inet6-mvpn signaling
```

```
user@PE1# set group IBGP family inet-mdt signaling
user@PE1# set group IBGP neighbor 10.255.162.100
```

7. Configure OSPF traffic engineering attributes and enable OSPF on the interfaces.

```
[edit protocols ospf]
user@PE1# set traffic-engineering
user@PE1# set area 0.0.0.1 interface fxp0.0 disable
user@PE1# set area 0.0.0.1 interface ge-3/3/2.0
user@PE1# set area 0.0.0.1 interface lo0.0
```

8. Enable LDP on all the interfaces and advertise P2MP capability to peers.

```
[edit protocols ldp]
user@PE1# set interface all
user@PE1# set p2mp
```

9. Configure PIM on the interfaces.

```
[edit protocols pim]
user@PE1# set interface all
user@PE1# set interface fxp0.0 disable
user@PE1# set interface lo0.0
user@PE1# set default-vpn-source interface-name lo0.0
```

10. Configure the routing policy.

```
[edit policy-options policy-statement]
user@PE1# set bgp-to-ospf from protocol bgp
user@PE1# set bgp-to-ospf then accept
```

11. Configure the routing instance type, interface, and the route distinguisher for the routing instance.

```
[edit routing-instances]
user@PE1# set vpn1 instance-type vrf
user@PE1# set vpn1 interface ge-2/0/10
```

```
user@PE1# set vpn1 interface lo0.201
user@PE1# set vpn1 route-distinguisher 10.255.162.109:100
```

12. Configure provider tunnel attributes for the routing instance.

```
[edit routing-instances]
user@PE1# set vpn1 provider-tunnel selective group 192.0.2.2/24 source 172.16.1.2/32
ingress-replication label-switched-path
user@PE1# set vpn1 provider-tunnel selective group 192.0.2.2/24 source 172.16.1.2/32
threshold-rate 10
user@PE1# set vpn1 provider-tunnel selective group 192.0.2.2/24 source 172.16.1.2/32 inter-
region-segmented threshold 0
user@PE1# set vpn1 provider-tunnel selective group 192.0.2.1/24 source 172.16.1.2/32
ingress-replication label-switched-path
user@PE1# set vpn1 provider-tunnel selective group 192.0.2.1/24 source 172.16.1.2/32
threshold-rate 0
user@PE1# set vpn1 provider-tunnel selective group 192.0.2.1/24 source 172.16.1.2/32 inter-
region-segmented threshold 10
user@PE1# set vpn1 provider-tunnel selective group 192.0.2.3/24 source 172.16.1.2/32
ingress-replication label-switched-path
user@PE1# set vpn1 provider-tunnel selective group 192.0.2.3/24 source 172.16.1.2/32
threshold-rate 0
user@PE1# set vpn1 provider-tunnel selective group 192.0.2.3/24 source 172.16.1.2/32 inter-
region-segmented threshold 0
user@PE1# set vpn1 provider-tunnel family inet ingress-replication label-switched-path
user@PE1# set vpn1 provider-tunnel family inet6 ingress-replication label-switched-path
```

13. Configure the VRF target community and advertise a single VPN label for all the routes in the VRF.

```
[edit routing-instances]
user@PE1# set vpn1 vrf-target target:123:1
user@PE1# set vpn1 vrf-table-label
```

14. Enable OSPF for the routing instance.

```
[edit routing-instances]
user@PE1# set vpn1 protocols ospf export bgp-to-ospf
user@PE1# set vpn1 protocols ospf area 0.0.0.1 interface all
user@PE1# set vpn1 protocols ospf area 0.0.0.1 interface lo0.201
```

15. Enable OSPF3 for the routing instance.

```
[edit routing-instances]
user@PE1# set vpn1 protocols ospf3 export bgp-to-ospf
user@PE1# set vpn1 protocols ospf3 area 0.0.0.1 interface all
```

16. Enable PIM attributes for the routing instance.

```
[edit routing-instances]
user@PE1# set vpn1 protocols pim dense-groups 192.0.2.39/24
user@PE1# set vpn1 protocols pim dense-groups 192.0.2.40/24
user@PE1# set vpn1 protocols pim rp local family inet address 198.51.100.1
user@PE1# set vpn1 protocols pim rp static address ::198.51.100.1
user@PE1# set vpn1 protocols pim interface all mode sparse-dense
```

Results

From configuration mode, confirm your configuration by entering the `show interfaces`, `show policy-options`, `show protocols`, `show routing-instances`, and `show routing-options` commands. If the output does not display the intended configuration, repeat the instructions in this example to correct the configuration.

```
user@PE1# show interfaces
ge-2/0/10 {
    unit 1 {
        family inet {
            address 192.0.2.2/24;
        }
        family inet6 {
            address ::192.0.2.2/120;
        }
        family mpls;
    }
}
ge-3/3/2 {
    unit 0 {
        family inet {
            address 203.0.113.1/24;
        }
        family iso;
```

```
        family inet6 {
            address ::203.0.113.1/120;
        }
        family mpls;
    }
}
lo0 {
    unit 201{
        family inet {
            address 198.51.100.1/24;
        }
    }
}
```

```
user@PE1# show policy-options
policy-statement bgp-to-ospf {
    from protocol bgp;
    then accept;
}
```

```
user@PE1# show protocols
rsvp {
    interface fxp0.0 {
        disable;
    }
    interface ge-3/3/2.0;
    interface lo0.0;
}
mpls {
    ipv6-tunneling;
    interface fxp0.0 {
        disable;
    }
    interface ge-3/3/2.0;
    interface lo0.0;
}
bgp {
    group IBGP {
        type internal;
        local-address 10.255.162.109;
```

```
        family inet {
            any;
        }
        family inet-vpn {
            unicast;
            multicast;
        }
        family inet6 {
            any;
        }
        family inet6-vpn {
            unicast;
        }
        family inet-mvpn {
            signaling;
        }
        family inet6-mvpn {
            signaling;
        }
        family inet-mdt {
            signaling;
        }
        neighbor 10.255.162.100;
    }
}
ospf {
    traffic-engineering;
    area 0.0.0.1 {
        interface fxp0.0 {
            disable;
        }
        interface ge-3/3/2.0;
        interface lo0.0;
    }
}
ldp {
    interface all;
    p2mp;
}
pim {
    interface all;
    interface fxp0.0 {
        disable;
```

```
    }
    interface lo0.0;
    default-vpn-source {
        interface-name lo0.0;
    }
}
```

```
user@PE1# show routing-instances
vpn1 {
    instance-type vrf;
    interface ge-2/0/10;
    interface lo0.201;
    route-distinguisher 10.255.162.109:100;
    provider-tunnel {
        selective {
            group 192.0.2.2/24 {
                source 172.16.1.2/32 {
                    ingress-replication {
                        label-switched-path;
                    }
                    threshold-rate 10;
                    inter-region-segmented {
                        threshold 0;
                    }
                }
            }
            group 192.0.2.1/24 {
                source 172.16.1.2/32 {
                    ingress-replication {
                        label-switched-path;
                    }
                    threshold-rate 0;
                    inter-region-segmented {
                        threshold 10;
                    }
                }
            }
            group 192.0.2.3/24 {
                source 172.16.1.2/32 {
                    ingress-replication {
                        label-switched-path;
```

```
                    }
                    threshold-rate 0;
                    inter-region-segmented {
                        threshold 0;
                    }
                }
            }
        }
        family {
            inet {
                ingress-replication {
                    label-switched-path;
                }
            }
            inet6 {
                ingress-replication {
                    label-switched-path;
                }
            }
        }
    }
    vrf-target target:123:1;
    vrf-table-label;
    protocols {
        ospf {
            export bgp-to-ospf;
            area 0.0.0.1 {
                interface all;
                interface lo0.201;
            }
        }
        ospf3 {
            export bgp-to-ospf;
            area 0.0.0.1 {
                interface all;
            }
        }
        pim {
            dense-groups {
                192.0.2.39/24;
                192.0.2.40/24;
            }
            rp {
```

```
                local {
                    family inet {
                        address 198.51.100.1;
                    }
                }
                static {
                    address ::198.51.100.1;
                }
            }
            interface all {
                mode sparse-dense;
            }
        }
    }
}
```

```
user@PE1# show routing-options
autonomous-system 65550;
```

Configuring ABR1

Step-by-Step Procedure

The following example requires that you navigate various levels in the configuration hierarchy. For information about navigating the CLI, see *Using the CLI Editor in Configuration Mode* in the CLI User Guide.

To configure Device ABR1:

1. Configure the interfaces.

```
[edit interfaces]
user@ABR1# set ge-1/0/4 unit 0 family inet address 203.0.113.12/24
user@ABR1# set ge-1/0/4 unit 0 family iso
user@ABR1# set ge-1/0/4 unit 0 family inet6 address ::203.0.113.12/120
user@ABR1# set ge-1/0/4 unit 0 family mpls
user@ABR1# set ge-1/1/3 unit 0 family inet address 203.0.113.5/24
user@ABR1# set ge-1/1/3 unit 0 family iso
user@ABR1# set ge-1/1/3 unit 0 family inet6 address ::203.0.113.5/120
user@ABR1# set ge-1/1/3 unit 0 family mpls
user@ABR1# set ge-1/1/9 unit 0 family inet address 203.0.113.4/24
```

```
user@ABR1# set ge-1/1/9 unit 0 family iso
user@ABR1# set ge-1/1/9 unit 0 family inet6 address ::203.0.113.4/120
user@ABR1# set ge-1/1/9 unit 0 family mpls
user@ABR1# set lo0 unit 0 family inet address 203.0.113.0/24
user@ABR1# set lo0 unit 0 family inet address 10.255.162.100/32 primary
```

2. Configure the autonomous system number.

```
[edit routing-options]
user@ABR1# set autonomous-system 65550
```

3. Disable RSVP on the management interface and enable RSVP on the interfaces.

```
[edit protocols rsvp]
user@ABR1# set interface fxp0.0 disable
user@ABR1# set interface ge-1/1/9.0
user@ABR1# set interface ge-1/0/4.0
user@ABR1# set interface ge-1/1/3.0
user@ABR1# set interface lo0.0
user@ABR1# set interface all
```

4. Configure MPLS IPv6 tunneling.

```
[edit protocols mpls]
user@ABR1# set ipv6-tunneling
```

5. Configure MPLS on the interfaces.

```
[edit protocols mpls]
user@ABR1# set interface fxp0.0 disable
user@ABR1# set interface ge-1/1/9.0
user@ABR1# set interface ge-1/0/4.0
user@ABR1# set interface ge-1/1/3.0
user@ABR1# set interface lo0.0
user@ABR1# set interface all
```

6. Configure the BGP protocol.

```
[edit protocols bgp]
user@ABR1# set group IBGP_1 type internal
user@ABR1# set group IBGP_1 local-address 10.255.162.100
user@ABR1# set group IBGP_1 family inet any
user@ABR1# set group IBGP_1 family inet-vpn unicast
user@ABR1# set group IBGP_1 family inet-vpn multicast
user@ABR1# set group IBGP_1 family inet6 any
user@ABR1# set group IBGP_1 family inet6-vpn unicast
user@ABR1# set group IBGP_1 family inet-mvpn signaling
user@ABR1# set group IBGP_1 family inet6-mvpn signaling
user@ABR1# set group IBGP_1 family inet-mdt signaling
user@ABR1# set group IBGP_1 cluster 0.0.0.1
user@ABR1# set group IBGP_1 neighbor 10.255.162.109
user@ABR1# set group IBGP_0 type internal
user@ABR1# set group IBGP_0 local-address 10.255.162.100
user@ABR1# set group IBGP_0 family inet any
user@ABR1# set group IBGP_0 family inet-vpn unicast
user@ABR1# set group IBGP_0 family inet-vpn multicast
user@ABR1# set group IBGP_0 family inet6 any
user@ABR1# set group IBGP_0 family inet6-vpn unicast
user@ABR1# set group IBGP_0 family inet-mvpn signaling
user@ABR1# set group IBGP_0 family inet6-mvpn signaling
user@ABR1# set group IBGP_0 family inet-mdt signaling
user@ABR1# set group IBGP_0 neighbor 10.255.162.117
user@ABR1# set group IBGP_0 neighbor 10.255.162.107
```

7. Configure OSPF traffic engineering attributes and enable OSPF on the interfaces.

```
[edit protocols ospf]
user@ABR1# set traffic-engineering
user@ABR1# set area 0.0.0.1 interface fxp0.0 disable
user@ABR1# set area 0.0.0.1 interface ge-1/1/9.0
user@ABR1# set area 0.0.0.0 interface ge-1/0/4.0
user@ABR1# set area 0.0.0.0 interface ge-1/1/3.0
user@ABR1# set area 0.0.0.0 interface lo0.0
```

8. Enable LDP on all the interfaces and advertise P2MP capability to peers.

```
[edit protocols ldp]
user@ABR1# set interface all
user@ABR1# set p2mp
```

9. Configure PIM on the interfaces.

```
[edit protocols pim]
user@ABR1# set interface all
user@ABR1# set interface fxp0.0 disable
user@ABR1# set interface lo0.0
```

10. Configure the tunnels of the inter-region template for a specific region or all regions.

```
[edit protocols mvpn inter-region-template]
user@ABR1# set template template_1 region IBGP_0 rsvp-te label-switched-path-template
default-template
user@ABR1# set template template_2 region IBGP_0 ldp-p2mp
user@ABR1# set template template_3 region IBGP_0 ingress-replication create-new-ucast-tunnel
user@ABR1# set template template_3 region IBGP_0 ingress-replication label-switched-path
label-switched-path-template default-template
user@ABR1# set template template_4 all-regions incoming
user@ABR1# set template template_5 region IBGP_0 rsvp-te static-lsp ABR1_to_ABR3
```

11. Configure the routing instance type, route distinguisher, inter-region template of the provider tunnel, and VRF target community, and advertise a single VPN label for all the routes in the VRF for the routing instance.

```
[edit routing-instances]
user@ABR1# set vpn1 instance-type vrf
user@ABR1# set vpn1 route-distinguisher 10.255.162.100:100
user@ABR1# set vpn1 provider-tunnel inter-region template template_1
user@ABR1# set vpn1 vrf-target target:123:1
user@ABR1# set vpn1 vrf-table-label
```

Results

From configuration mode, confirm your configuration by entering the `show interfaces`, `show protocols`, `show routing-instances`, and `show routing-options` commands. If the output does not display the intended configuration, repeat the instructions in this example to correct the configuration.

```
user@ABR1# show interfaces
ge-1/0/4 {
    unit 0 {
        family inet {
            address 203.0.113.12/24;
        }
        family iso;
        family inet6 {
            address ::203.0.113.12/120;
        }
        family mpls;
    }
}
ge-1/1/3 {
    unit 0 {
        family inet {
            address 203.0.113.5/24;
        }
        family iso;
        family inet6 {
            address ::203.0.113.5/120;
        }
        family mpls;
    }
}
ge-1/1/9 {
    unit 0 {
        family inet {
            address 203.0.113.4/24;
        }
        family iso;
        family inet6 {
            address ::203.0.113.4/120;
        }
        family mpls;
    }
```

```
}
lo0 {
    unit 201 {
        family inet {
            address 203.0.113.0/24;
            address 10.255.162.100/32 {
                primary;
            }
        }
    }
}
```

```
user@ABR1# show protocols
rsvp {
    interface fxp0.0 {
        disable;
    }
    interface ge-1/1/9.0;
    interface ge-1/0/4.0;
    interface ge-1/1/3.0;
    interface lo0.0;
    interface all;
}
mpls {
    ipv6-tunneling;
    interface fxp0.0 {
        disable;
    }
    interface ge-1/1/9.0;
    interface ge-1/0/4.0;
    interface ge-1/1/3.0;
    interface lo0.0;
    interface all;
}
bgp {
    group IBGP_1 {
        type internal;
        local-address 10.255.162.100;
        family inet {
            any;
        }
```

```
        family inet-vpn {
            unicast;
            multicast;
        }
        family inet6 {
            any;
        }
        family inet6-vpn {
            unicast;
        }
        family inet-mvpn {
            signaling;
        }
        family inet6-mvpn {
            signaling;
        }
        family inet-mdt {
            signaling;
        }
        cluster 0.0.0.1;
        neighbor 10.255.162.109;
    }
    group IBGP_0 {
        type internal;
        local-address 10.255.162.100;
        family inet {
            any;
        }
        family inet-vpn {
            unicast;
            multicast;
        }
        family inet6 {
            any;
        }
        family inet6-vpn {
            unicast;
        }
        family inet-mvpn {
            signaling;
        }
        family inet6-mvpn {
            signaling;
```

```
        }
        family inet-mdt {
            signaling;
        }
        neighbor 10.255.162.117;
        neighbor 10.255.162.107;
    }
}
ospf {
    traffic-engineering;
    area 0.0.0.1 {
        interface fxp0.0 {
            disable;
        }
        interface ge-1/1/9.0;
    }
    area 0.0.0.0 {
        interface ge-1/0/4.0;
        interface ge-1/1/3.0;
        interface lo0.0;
    }
}
ldp {
    interface all;
    p2mp;
}
pim {
    interface all;
    interface fxp0.0 {
        disable;
    }
    interface lo0.0;
}
mvpn {
    inter-region-template {
        template template_1 {
            region IBGP_0 {
                rsvp-te {
                    label-switched-path-template {
                        default-template;
                    }
                }
            }
```

```
        }
        template template_2 {
            region IBGP_0 {
                ldp-p2mp;
            }
        }
        template template_3 {
            region IBGP_0 {
                ingress-replication {
                    create-new-ucast-tunnel;
                    label-switched-path {
                        label-switched-path-template {
                            default-template;
                        }
                    }
                }
            }
        }
        template template_4 {
            all-regions {
                incoming;
            }
        }
        template template_5 {
            region IBGP_0 {
                rsvp-te {
                    static-lsp ABR1_to_ABR3;
                }
            }
        }
    }
}
```

```
user@ABR1# show routing-instances
vpn1 {
    instance-type vrf;
    route-distinguisher 10.255.162.100:100;
    provider-tunnel {
        inter-region {
            template template_1;
        }
```

```
    }
    vrf-target target:123:1;
    vrf-table-label;
}
```

```
user@ABR1# show routing-options
autonomous-system 65550;
```

Configuring ABR2

Step-by-Step Procedure

The following example requires that you navigate various levels in the configuration hierarchy. For information about navigating the CLI, see *Using the CLI Editor in Configuration Mode* in the CLI User Guide.

To configure Device ABR2:

1. Configure the interfaces.

```
[edit interfaces]
user@ABR2# set ge-1/0/3 unit 0 family inet address 203.0.113.11/24
user@ABR2# set ge-1/0/3 unit 0 family iso
user@ABR2# set ge-1/0/3 unit 0 family inet6 address ::203.0.113.11/120
user@ABR2# set ge-1/0/3 unit 0 family mpls
user@ABR2# set ge-1/1/4 unit 0 family inet address 203.0.113.10/24
user@ABR2# set ge-1/1/4 unit 0 family iso
user@ABR2# set ge-1/1/4 unit 0 family inet6 address ::203.0.113.10/120
user@ABR2# set ge-1/1/4 unit 0 family mpls
user@ABR2# set ge-1/1/10 unit 1 family inet address 192.0.2.2/24
user@ABR2# set ge-1/1/10 unit 1 family inet6 address ::192.0.2.2/120
user@ABR2# set ge-1/1/10 unit 1 family mpls
```

2. Configure the autonomous system number.

```
[edit routing-options]
user@ABR2# set autonomous-system 65550
```

3. Disable RSVP on the management interface and enable RSVP on the interfaces.

```
[edit protocols rsvp]
user@ABR2# set interface fxp0.0 disable
user@ABR2# set interface lo0.0
user@ABR2# set interface all
```

4. Enable MPLS IPv6 tunneling.

```
[edit protocols mpls]
user@ABR2# set ipv6-tunneling
```

5. Disable MPLS on the management interface and enable RSVP on the interfaces.

```
[edit protocols mpls]
user@ABR2# set interface fxp0.0 disable
user@ABR2# set interface lo0.0
user@ABR2# set interface all
```

6. Configure the BGP protocol.

```
[edit protocols bgp]
user@ABR2# set group IBGP_2 type internal
user@ABR2# set group IBGP_2 local-address 10.255.162.117
user@ABR2# set group IBGP_2 family inet any
user@ABR2# set group IBGP_2 family inet-vpn unicast
user@ABR2# set group IBGP_2 family inet-vpn multicast
user@ABR2# set group IBGP_2 family inet6 any
user@ABR2# set group IBGP_2 family inet6-vpn unicast
user@ABR2# set group IBGP_2 family inet-mvpn signaling
user@ABR2# set group IBGP_2 family inet6-mvpn signaling
user@ABR2# set group IBGP_2 family inet-mdt signaling
user@ABR2# set group IBGP_2 cluster 0.0.0.2
user@ABR2# set group IBGP_2 neighbor 10.255.162.104
user@ABR2# set group IBGP_2 neighbor 198.51.100.17
user@ABR2# set group IBGP_0 type internal
user@ABR2# set group IBGP_0 local-address 10.255.162.117
user@ABR2# set group IBGP_0 family inet any
user@ABR2# set group IBGP_0 family inet-vpn unicast
```

```
user@ABR2# set group IBGP_0 family inet-vpn multicast
user@ABR2# set group IBGP_0 family inet6 any
user@ABR2# set group IBGP_0 family inet6-vpn unicast
user@ABR2# set group IBGP_0 family inet-mvpn signaling
user@ABR2# set group IBGP_0 family inet6-mvpn signaling
user@ABR2# set group IBGP_0 family inet-mdt signaling
user@ABR2# set group IBGP_0 neighbor 10.255.162.100
user@ABR2# set group IBGP_0 neighbor 10.255.162.107
```

7. Configure OSPF traffic engineering attributes, and disable OSPF on the management interface and enable OSPF on the interfaces.

```
[edit protocols ospf]
user@ABR2# set traffic-engineering
user@ABR2# set area 0.0.0.0 interface fxp0.0 disable
user@ABR2# set area 0.0.0.0 interface ge-1/0/3.0
user@ABR2# set area 0.0.0.0 interface ge-1/1/4.0
user@ABR2# set area 0.0.0.0 interface lo0.0
user@ABR2# set area 0.0.0.2 interface ge-1/1/10.1
```

8. Enable LDP on all the interfaces and advertise P2MP capability to peers.

```
[edit protocols ldp]
user@ABR2# set interface all
user@ABR2# set p2mp
```

9. Configure PIM on the interfaces.

```
[edit protocols pim]
user@ABR2# set interface fxp0.0 all
user@ABR2# set interface fxp0.0 disable
user@ABR2# set interface lo0.0
```

10. Configure the tunnels of the inter-region template for a specific region or all regions.

```
[edit protocols mvpn inter-region-template]
user@ABR2# set template template_1 region IBGP_2 rsvp-te label-switched-path-template
default-template
user@ABR2# set template template_2 region IBGP_2 ldp-p2mp
```

```
user@ABR2# set template template_3 region IBGP_2 ingress-replication create-new-ucast-tunnel
user@ABR2# set template template_3 region IBGP_2 ingress-replication label-switched-path
label-switched-path-template default-template
user@ABR2# set template template_4 all-regions incoming
user@ABR2# set template template_5 region IBGP_2 rsvp-te static-lsp ABR2_to_PE2_3
```

11. Configure the routing instance type, route distinguisher, inter-region template of the provider tunnel, and VRF target community, and advertise a single VPN label for all the routes in the VRF for the routing instance.

```
[edit routing-instances]
user@ABR2# set vpn1 instance-type vrf
user@ABR2# set vpn1 route-distinguisher 10.255.162.117:100
user@ABR2# set vpn1 provider-tunnel inter-region template template_1
user@ABR2# set vpn1 vrf-target target:123:1
user@ABR2# set vpn1 vrf-table-label
```

Results

```
user@ABR2# show interfaces
ge-1/0/3 {
    unit 0 {
        family inet {
            address 203.0.113.11/24;
        }
        family iso;
        family inet6 {
            address ::203.0.113.11/120;
        }
        family mpls;
    }
}
ge-1/1/4 {
    unit 0 {
        family inet {
            address 203.0.113.10/24;
        }
        family iso;
        family inet6 {
            address ::203.0.113.10/120;
```

```
        }
        family mpls;
    }
}
ge-1/1/10 {
    unit 1 {
        family inet {
            address 192.0.2.2/24;
        }
        family inet6 {
            address ::192.0.2.2/120;
        }
        family mpls;
    }
}
lo0 {
    unit 201 {
        family inet {
            address 203.0.113.0/24;
            address 10.255.162.117/32  {
                primary;
            }
        }
    }
}
```

```
user@ABR2# show protocols
rsvp {
    interface fxp0.0 {
        disable;
    }
    interface lo0.0;
    interface all;
}
mpls {
    ipv6-tunneling;
    interface fxp0.0 {
        disable;
    }
    interface lo0.0;
    interface all;
```

```
}
bgp {
    group IBGP_2 {
        type internal;
    local-address 10.255.162.117;
        family inet {
            any;
        }
        family inet-vpn {
            unicast;
            multicast;
        }
        family inet6 {
            any;
        }
        family inet6-vpn {
            unicast;
        }
        family inet-mvpn {
            signaling;
        }
        family inet6-mvpn {
            signaling;
        }
        family inet-mdt {
            signaling;
            }
            cluster 0.0.0.2;
            neighbor 10.255.162.104;
            neighbor 198.51.100.17;
        }
        group IBGP_0 {
            type internal;
            local-address 10.255.162.117;
            family inet {
                any;
            }
            family inet-vpn {
                unicast;
                multicast;
            }
            family inet6 {
                any;
```

```
        }
        family inet6-vpn {
            unicast;
        }
        family inet-mvpn {
            signaling;
        }
        family inet6-mvpn {
            signaling;
        }
        family inet6-mdt {
            signaling;
        }
        neighbor 10.255.162.100;
        neighbor 10.255.162.107;
    }
    ospf {
        traffic-engineering;
        area 0.0.0.0 {
            interface fxp0.0 {
                disable;
            }
            interface ge-1/0/3.0;
            interface ge-1/1/4.0;
            interface lo0.0;
        }
        area 0.0.0.2 {
            interface ge-1/1/10.1;
        }
    }
    ldp {
        interface all;
        p2mp;
    }
    pim {
        interface all;
        interface fxp0.0 {
            disable;
        }
        interface lo0.0;
    }
    mvpn {
        inter-region-template {
```

```
                template template_1 {
                    region IBGP_2 {
                        rsvp-te {
                            label-switched-path-template {
                                default-template;
                            }
                        }
                    }
                }
                template template_2 {
                    region IBGP_2 {
                        ldp-p2mp;
                    }
                }
                template template_3 {
                    region IBGP_2 {
                        ingress-replication {
                            create-new-ucast-tunnel;
                            label-switched-path {
                                label-switched-path-template {
                                    default-template;
                                }
                            }
                        }
                    }
                }
                template template_4 {
                    all-regions {
                        incoming;
                    }
                }
                template template_5 {
                    region IBGP_2 {
                        rsvp-te {
                            static-lsp ABR2_to_PE2_3;
                        }
                    }
                }
```

```
            }
        }
```

```
user@ABR2# show routing-instances
vpn1 {
    instance-type vrf;
    route-distinguisher 10.255.162.100:100;
    provider-tunnel {
        inter-region {
            template template_1;
        }
    }
    vrf-target target:123:1;
    vrf-table-label;
}
```

```
user@ABR2# show routing-options
autonomous-system 65550;
```

Configuring ABR3

Step-by-Step Procedure

The following example requires that you navigate various levels in the configuration hierarchy. For information about navigating the CLI, see *Using the CLI Editor in Configuration Mode* in the CLI User Guide.

To configure Device ABR3:

1. Configure the interfaces.

   ```
   [edit interfaces]
   user@ABR3# set ge-1/0/8 unit 0 family inet address 203.0.113.6/24
   user@ABR3# set ge-1/0/8 unit 0 family iso
   user@ABR3# set ge-1/0/8 unit 0 family inet6 address ::203.0.113.6/120
   user@ABR3# set ge-1/0/8 unit 0 family mpls
   user@ABR3# set ge-1/1/4 unit 0 family inet address 203.0.113.9/24
   user@ABR3# set ge-1/1/4 unit 0 family iso
   user@ABR3# set ge-1/1/4 unit 0 family inet6 address ::203.0.113.9/120
   ```

```
user@ABR3# set ge-1/1/4 unit 0 family mpls
user@ABR3# set ge-1/3/1 unit 0 family inet address 203.0.113.8/24
user@ABR3# set ge-1/3/1 unit 0 family iso
user@ABR3# set ge-1/3/1 unit 0 family inet6 address ::203.0.113.8/120
user@ABR3# set ge-1/3/1 unit 0 family mpls
user@ABR3# set lo0 unit 0 family inet address 203.0.113.0/24
user@ABR3# set lo0 unit 0 family inet address 10.255.162.107/32 primary
```

2. Configure the autonomous system number.

```
[edit routing-options]
user@ABR3# set autonomous-system 65550
```

3. Configure RSVP on all the interfaces, excluding the management interface.

```
[edit protocols rsvp]
user@ABR3# set interface all
user@ABR3# set interface fxp0.0 disable
user@ABR3# set interface lo0.0
```

4. Configure MPLS IPv6 tunneling, configure the label-switched path, and enable MPLS on all the interfaces, excluding the management interface.

```
[edit protocols mpls]
user@ABR3# set ipv6-tunneling
user@ABR3# set label-switched-path ABR3_to_PE3 from 10.255.162.107
user@ABR3# set label-switched-path ABR3_to_PE3 to 10.255.162.102
user@ABR3# set label-switched-path ABR3_to_PE3 p2mp vpn1
user@ABR3# set label-switched-path ABR3_to_ABR1 from 10.255.162.107
user@ABR3# set label-switched-path ABR3_to_ABR1 to 10.255.162.100
user@ABR3# set label-switched-path ABR3_to_ABR1 p2mp vpn1
user@ABR3# set label-switched-path ABR3_to_ABR2 from 10.255.162.107
user@ABR3# set label-switched-path ABR3_to_ABR2 to 10.255.162.117
user@ABR3# set label-switched-path ABR3_to_ABR2 p2mp vpn1
user@ABR3# set interface all
user@ABR3# set interface fxp0.0 disable
user@ABR3# set interface lo0.0
```

5. Configure the BGP protocol.

```
[edit protocols bgp]
user@ABR3# set group IBGP_3 type internal
user@ABR3# set group IBGP_3 local-address 10.255.162.107
user@ABR3# set group IBGP_3 family inet any
user@ABR3# set group IBGP_3 family inet-vpn unicast
user@ABR3# set group IBGP_3 family inet-vpn multicast
user@ABR3# set group IBGP_3 family inet6 any
user@ABR3# set group IBGP_3 family inet6-vpn unicast
user@ABR3# set group IBGP_3 family inet-mvpn signaling
user@ABR3# set group IBGP_3 family inet6-mvpn signaling
user@ABR3# set group IBGP_3 family inet-mdt signaling
user@ABR3# set group IBGP_3 cluster 0.0.0.3
user@ABR3# set group IBGP_3 neighbor 10.255.162.102
user@ABR3# set group IBGP_0 type internal
user@ABR3# set group IBGP_0 local-address 10.255.162.107
user@ABR3# set group IBGP_0 family inet any
user@ABR3# set group IBGP_0 family inet-vpn unicast
user@ABR3# set group IBGP_0 family inet-vpn multicast
user@ABR3# set group IBGP_0 family inet6 any
user@ABR3# set group IBGP_0 family inet6-vpn unicast
user@ABR3# set group IBGP_0 family inet-mvpn signaling
user@ABR3# set group IBGP_0 family inet6-mvpn signaling
user@ABR3# set group IBGP_0 family inet-mdt signaling
user@ABR3# set group IBGP_0 neighbor 10.255.162.100
user@ABR3# set group IBGP_0 neighbor 10.255.162.117
```

6. Configure OSPF traffic engineering attributes, disable OSPF on the management interface, and enable OSPF on the interfaces.

```
[edit protocols ospf]
user@ABR3# set traffic-engineering
user@ABR3# set area 0.0.0.0 interface fxp0.0 disable
user@ABR3# set area 0.0.0.0 interface ge-1/0/8.0
user@ABR3# set area 0.0.0.0 interface ge-1/1/4.0
user@ABR3# set area 0.0.0.0 interface lo0.0
user@ABR3# set area 0.0.0.3 interface ge-1/3/1.0
```

7. Enable LDP on all the interfaces and advertise P2MP capability to peers.

```
[edit protocols ldp]
user@ABR3# set interface all
user@ABR3# set p2mp
```

8. Configure PIM on the interfaces.

```
[edit protocols pim]
user@ABR3# set interface all
user@ABR3# set interface fxp0.0 disable
user@ABR3# set interface lo0.0
```

9. Configure the tunnels of the inter-region template for a specific region or all regions.

```
[edit protocols mvpn inter-region-template]
user@ABR3# set template template_1 region IBGP_3 rsvp-te label-switched-path-template
default-template
user@ABR3# set template template_2 region IBGP_3 ldp-p2mp
user@ABR3# set template template_3 region IBGP_3 ingress-replication create-new-ucast-tunnel
user@ABR3# set template template_3 region IBGP_3 ingress-replication label-switched-path
label-switched-path-template default-template
user@ABR3# set template template_4 all-regions incoming
user@ABR3# set template template_5 region IBGP_3 rsvp-te static-lsp ABR3_to_PE3
```

10. Configure the routing instance type, route distinguisher, inter-region template of the provider tunnel, and VRF target community, and advertise a single VPN label for all the routes in the VRF for the routing instance.

```
[edit routing-instances]
user@ABR3# set vpn1 instance-type vrf
user@ABR3# set vpn1 route-distinguisher 10.255.162.107:100
user@ABR3# set vpn1 provider-tunnel inter-region template template_1
user@ABR3# set vpn1 vrf-target target:123:1
user@ABR3# set vpn1 vrf-table-label
```

Results

From configuration mode, confirm your configuration by entering the `show interfaces`, `show policy-options`, `show protocols`, `show routing-instances`, and `show routing-options` commands. If the output does not display the intended configuration, repeat the instructions in this example to correct the configuration.

```
user@ABR3# show interfaces
ge-1/0/8 {
    unit 0 {
        family inet {
            address 203.0.113.6/24;
        }
        family iso;
        family inet6 {
            address ::203.0.113.6/120;
        }
        family mpls;
    }
}
ge-1/1/4 {
    unit 0 {
        family inet {
            address 203.0.113.9/24;
        }
        family iso;
        family inet6 {
            address ::203.0.113.9/120;
        }
        family mpls;
    }
}
ge-1/3/1 {
    unit 0 {
        family inet {
            address 203.0.113.8/24;
        }
        family iso;
        family inet6 {
            address ::203.0.113.8/120;
        }
        family mpls;
    }
```

```
}
lo0 {
    unit 0 {
        family inet {
            address 203.0.113.0/24;
            address 10.255.162.107/32 {
                primary;
            }
        }
    }
}
```

```
user@ABR3# show protocols
rsvp {
    interface all;
    interface fxp0.0 {
        disable;
    }
    interface lo0.0;
}
mpls {
    ipv6-tunneling;
    label-switched-path ABR3_to_PE3{
        from 10.255.162.107;
        to 10.255.162.102;
        p2mp vpn1;
    }
    label-switched-path ABR3_to_ABR1 {
        from 10.255.162.107;
        to 10.255.162.100;
        p2mp vpn1;
    }
    label-switched-path ABR3_to_ABR2 {
        from 10.255.162.107;
        to 10.255.162.117;
        p2mp vpn1;
    }
    interface all;
    interface fxp0.0 {
        disable;
    }
```

```
    interface lo0.0;
}
bgp {
    group IBGP_3 {
        type internal;
        local-address 10.255.162.107;
        family inet {
            any;
        }
        family inet-vpn {
            unicast;
            multicast;
        }
        family inet6 {
            any;
        }
        family inet6-vpn {
            unicast;
        }
        family inet-mvpn {
            signaling;
        }
        family inet6-mvpn {
            signaling;
        }
        family inet-mdt {
            signaling;
        }
        cluster 0.0.0.3;
        neighbor 10.255.162.102;
    }
    group IBGP_0 {
        type internal;
        local-address 10.255.162.107;
        family inet {
            any;
        }
        family inet-vpn {
            unicast;
            multicast;
        }
        family inet6 {
            any;
```

```
        }
        family inet6-vpn {
            unicast;
        }
        family inet-mvpn {
            signaling;
        }
        family inet6-mvpn {
            signaling;
        }
        family inet-mdt {
            signaling;
        }
        neighbor 10.255.162.100;
        neighbor 10.255.162.117;
    }
}
ospf {
    traffic-engineering;
    area 0.0.0.0 {
        interface fxp0.0 {
            disable;
        }
        interface ge-1/0/8.0;
        interface ge-1/1/4.0;
        interface lo0.0;
    }
    area 0.0.0.3 {
        interface ge-1/3/1.0;
    }
}
ldp {
    interface all;
    p2mp;
}
pim {
    interface all;
    interface fxp0.0 {
        disable;
    }
    interface lo0.0;
}
mvpn {
```

```
    inter-region-template {
        template template_1 {
            region IBGP_3 {
                rsvp-te {
                    label-switched-path-template {
                        default-template;
                    }
                }
            }
        }
        template template_2 {
            region IBGP_3 {
                ldp-p2mp;
            }
        }
        template template_3 {
            region IBGP_3 {
                ingress-replication {
                    create-new-ucast-tunnel;
                    label-switched-path {
                        label-switched-path-template {
                            default-template;
                        }
                    }
                }
            }
        }
        template template_4 {
            all-regions {
                incoming;
            }
        }
        template template_5 {
            region IBGP_3 {
                rsvp-te {
                    static-lsp ABR3_to_PE3_1;
                }
            }
        }
```

```
    }
}
```

```
user@ABR3# show routing-instances
vpn1 {
    instance-type vrf;
    route-distinguisher 10.255.162.107:100;
    provider-tunnel {
        inter-region {
            template template_1;
        }
    }
    vrf-target target:123:1;
    vrf-table-label;
}
```

```
user@ABR3# show routing-option
autonomous-system 65550;
```

Verification

IN THIS SECTION

Confirm that the configuration is working properly.

Verifying Inflow at the Ingress PE Router

Purpose

Verify the traffic inflow into the ingress PE router for the given routing instance.

Action

From operational mode, run the `show multicast route extensive instance vpn1` command for Device PE1.

```
user@PE1>  show multicast route extensive instance vpn1
display-tunnel-name
Instance: vpn1 Family: INET

Group: 192.0.2.2
    Source: 172.16.1.2/32
    Upstream interface: ge-2/0/10.1
    Downstream interface list:
        mvpn:2
    Number of outgoing interfaces: 1
    Session description: Unknown
    Statistics: 3002 kBps, 10008 pps, 34124622 packets
    Next-hop ID: 0
    Upstream protocol: MVPN
    Route state: Active
    Forwarding state: Pruned
    Cache lifetime/timeout: forever
    Wrong incoming interface notifications: 0
    Uptime: 00:56:53

Group: 192.0.2.1
    Source: 172.16.1.2/32
    Upstream interface: ge-2/0/10.1
    Downstream interface list:
        mvpn:4
    Number of outgoing interfaces: 1
    Session description: Unknown
    Statistics: 3002 kBps, 10008 pps, 34125577 packets
    Next-hop ID: 0
    Upstream protocol: MVPN
    Route state: Active
    Forwarding state: Pruned
    Cache lifetime/timeout: forever
    Wrong incoming interface notifications: 0
    Uptime: 00:56:53

Group: 192.0.2.3
    Source: 172.16.1.2/32
```

```
    Upstream interface: ge-2/0/10.1
    Downstream interface list:
        mvpn:3
    Number of outgoing interfaces: 1
    Session description: Unknown
    Statistics: 3002 kBps, 10008 pps, 34124620 packets
    Next-hop ID: 0
    Upstream protocol: MVPN
    Route state: Active
    Forwarding state: Pruned
    Cache lifetime/timeout: forever
    Wrong incoming interface notifications: 0
    Uptime: 00:56:53
```

Meaning

The output shows the traffic inflow into the ingress Device PE1.

Verifying the Route Table for Segmented Type-3 Traffic Generated from Device ABR1 Toward PE1 Router

Purpose

Verify the route table for segmented Type-3 traffic generated from Device ABR1.

Action

From operational mode, run the `show route table vpn1.mvpn.0 match-prefix 3:* detail` command.

```
user@PE1> show route table vpn1.mvpn.0 match-prefix 3:* detail

vpn1.mvpn.0: 19 destinations, 22 routes (19 active, 3 holddown, 0 hidden)
3:10.255.162.109:100:32:172.16.1.2:32:20192.0.2.2:10.255.162.109/240 (1 entry, 1 announced)
        *MVPN   Preference: 70
                PMSI: Flags 0x1: Label 0: Type INGRESS-REPLICATION 10.255.162.109
                Next hop type: Indirect, Next hop index: 0
                Address: 0xa5b8690
                Next-hop reference count: 11
                Protocol next hop: 10.255.162.109
                Indirect next hop: 0x0 - INH Session ID: 0x0
                State: <Active Int Ext>
```

```
                Age: 1:00:20    Metric2: 1
                Validation State: unverified
                Task: mvpn global task
                Announcement bits (3): 0-PIM.vpn1 1-mvpn global task 2-rt-export
                AS path: I
                Communities: segmented-nh:10.255.162.109:0

3:10.255.162.109:100:32:172.16.1.2:32:20192.0.2.1:10.255.162.109/240 (1 entry, 1 announced)
        *MVPN   Preference: 70
                PMSI: Flags 0x1: Label 0: Type INGRESS-REPLICATION 10.255.162.109
                Next hop type: Indirect, Next hop index: 0
                Address: 0xa5b8690
                Next-hop reference count: 11
                Protocol next hop: 10.255.162.109
                Indirect next hop: 0x0 - INH Session ID: 0x0
                State: <Active Int Ext>
                Age: 59:50      Metric2: 1
                Validation State: unverified
                Task: mvpn global task
                Announcement bits (3): 0-PIM.vpn1 1-mvpn global task 2-rt-export
                AS path: I
                Communities: segmented-nh:10.255.162.109:0

3:10.255.162.109:100:32:172.16.1.2:32:192.0.2.3:10.255.162.109/240 (1 entry, 1 announced)
        *MVPN   Preference: 70
                PMSI: Flags 0x1: Label 0: Type INGRESS-REPLICATION 10.255.162.109
                Next hop type: Indirect, Next hop index: 0
                Address: 0xa5b8690
                Next-hop reference count: 11
                Protocol next hop: 10.255.162.109
                Indirect next hop: 0x0 - INH Session ID: 0x0
                State: <Active Int Ext>
                Age: 1:00:20    Metric2: 1
                Validation State: unverified
                Task: mvpn global task
                Announcement bits (3): 0-PIM.vpn1 1-mvpn global task 2-rt-export
                AS path: I
                Communities: segmented-nh:10.255.162.109:0
```

Meaning

The output indicates the route table for the segmented type-3 traffic generated from ABR1.

Verifying the Route Table for Segmented Type-4 Traffic Received from Device ABR1 Toward PE1 Router

Purpose

Verify the route table for segmented type-4 traffic received from Device ABR1.

Action

From operational mode, run the `show route table vpn1.mvpn.0 match-prefix 4:* detail` command.

```
user@PE1> show route table vpn1.mvpn.0 match-prefix 4:* detail

vpn1.mvpn.0: 19 destinations, 22 routes (19 active, 3 holddown, 0 hidden)
4:3:10.255.162.109:100:32:172.16.1.2:32:20192.0.2.2:10.255.162.109:10.255.162.100/240 (1 entry,
1 announced)
        *BGP    Preference: 170/-101
                PMSI: Flags 0x0: Label 300320: Type INGRESS-REPLICATION 10.255.162.100
                Next hop type: Indirect, Next hop index: 0
                Address: 0xa5d11d0
                Next-hop reference count: 24
                Source: 10.255.162.100
                Protocol next hop: 10.255.162.100
                Indirect next hop: 0x0 - INH Session ID: 0x0
                State: <Secondary Active Int Ext>
                Local AS: 65550 Peer AS: 65550
                Age: 1:00:29    Metric2: 2
                Validation State: unverified
                Task: BGP_65550.10.255.162.100
                Announcement bits (2): 0-PIM.vpn1 1-mvpn global task
                AS path: I
                Communities: target:10.255.162.109:0
                Import Accepted
                Localpref: 100
                Router ID: 10.255.162.100
                Primary Routing Table bgp.mvpn.0

4:3:10.255.162.109:100:32:172.16.1.2:32:20192.0.2.1:10.255.162.109:10.255.162.100/240 (1 entry,
1 announced)
        *BGP    Preference: 170/-101
                PMSI: Flags 0x0: Label 300352: Type INGRESS-REPLICATION 10.255.162.100
                Next hop type: Indirect, Next hop index: 0
```

```
                Address: 0xa5d11d0
                Next-hop reference count: 24
                Source: 10.255.162.100
                Protocol next hop: 10.255.162.100
                Indirect next hop: 0x0 - INH Session ID: 0x0
                State: <Secondary Active Int Ext>
                Local AS: 65550 Peer AS: 65550
                Age: 59:59      Metric2: 2
                Validation State: unverified
                Task: BGP_65550.10.255.162.100
                Announcement bits (2): 0-PIM.vpn1 1-mvpn global task
                AS path: I
                Communities: target:10.255.162.109:0
                Import Accepted
                Localpref: 100
                Router ID: 10.255.162.100
                Primary Routing Table bgp.mvpn.0

4:3:10.255.162.109:100:32:172.16.1.2:32:192.0.2.3:10.255.162.109:10.255.162.100/240 (1 entry, 1 
announced)
        *BGP    Preference: 170/-101
                PMSI: Flags 0x0: Label 300336: Type INGRESS-REPLICATION 10.255.162.100
                Next hop type: Indirect, Next hop index: 0
                Address: 0xa5d11d0
                Next-hop reference count: 24
                Source: 10.255.162.100
                Protocol next hop: 10.255.162.100
                Indirect next hop: 0x0 - INH Session ID: 0x0
                State: <Secondary Active Int Ext>
                Local AS: 65550 Peer AS: 65550
                Age: 1:00:29    Metric2: 2
                Validation State: unverified
                Task: BGP_65550.10.255.162.100
                Announcement bits (2): 0-PIM.vpn1 1-mvpn global task
                AS path: I
                Communities: target:10.255.162.109:0
                Import Accepted
                Localpref: 100
                Router ID: 10.255.162.100
                Primary Routing Table bgp.mvpn.0
```

Meaning

The output shows the route table for segmented type-4 traffic received from device ABR1.

Verifying the LDP Traffic Statistics

Purpose

Verify the LDP traffic statistics of Device PE1.

Action

From operational mode, run the `show ldp traffic-statistics` command.

```
user@PE1> show ldp traffic-statistics

INET FEC Statistics:

FEC                    Type                  Packets                Bytes    Shared
 10.255.162.100/32     Transit                     0                    0    No
                       Ingress             112882983          33864894900    No
 10.255.162.102/32     Transit                     0                    0    No
                       Ingress               3884115           1165234500    No
 10.255.162.104/32     Transit                     0                    0    No
                       Ingress               3884115           1165234500    No
 10.255.162.107/32     Transit                     0                    0    No
                       Ingress                     0                    0    No
 10.255.162.117/32     Transit                     0                    0    No
                       Ingress                     0                    0    No
 10.255.162.119/32     Transit                     0                    0    No
                       Ingress                     0                    0    No
 198.51.100.19/24       Transit                     0                    0    No
                       Ingress                     0                    0    No
 198.51.100.17/24       Transit                     0                    0    No
                       Ingress               3884115           1165234500    No
```

Meaning

The output shows the LDP traffic statistics.

Verification

Confirm that the configuration is working properly.

Verifying the Segmented Type-3 Traffic Received from the PE1 Router on ABR1 with the Tunnel-Type as IR

Purpose

Display the segmented type-3 traffic received from the PE1 router on ABR1 with the tunnel-type as IR.

Action

From operational mode, run the `show route table vpn1.mvpn.0 match-prefix 3:* detail` command.

```
user@ABR1> show route table vpn1.mvpn.0 match-prefix 3:* detail

vpn1.mvpn.0: 22 destinations, 22 routes (22 active, 0 holddown, 0 hidden)
3:10.255.162.109:100:32:172.16.1.2:32:192.0.2.2:10.255.162.109/240 (1 entry, 1 announced)
        *BGP    Preference: 170/-101
                PMSI: Flags 0x1: Label 0: Type INGRESS-REPLICATION 10.255.162.109
                Next hop type: Indirect, Next hop index: 0
                Address: 0xa5cddb0
                Next-hop reference count: 24
                Source: 10.255.162.109
                Protocol next hop: 10.255.162.109
                Indirect next hop: 0x0 - INH Session ID: 0x0
                State: <Secondary Active Int Ext>

                Local AS: 65550 Peer AS: 65550
                Age: 1:02:45    Metric2: 2
                Validation State: unverified
                Task: BGP_65550.10.255.162.109
                Announcement bits (1): 0-mvpn global task
```

```
                AS path: I
                Communities: target:123:1 segmented-nh:10.255.162.109:0
                Import Accepted
                Localpref: 100
                Router ID: 10.255.162.109
                Primary Routing Table bgp.mvpn.0

3:10.255.162.109:100:32:172.16.1.2:32:192.0.2.1:10.255.162.109/240 (1 entry, 1 announced)
        *BGP    Preference: 170/-101
                PMSI: Flags 0x1: Label 0: Type INGRESS-REPLICATION 10.255.162.109
                Next hop type: Indirect, Next hop index: 0
                Address: 0xa5cddb0
                Next-hop reference count: 24
                Source: 10.255.162.109
                Protocol next hop: 10.255.162.109
                Indirect next hop: 0x0 - INH Session ID: 0x0
                State: <Secondary Active Int Ext>

                Local AS: 65550 Peer AS: 65550
                Age: 1:02:15    Metric2: 2
                Validation State: unverified
                Task: BGP_65550.10.255.162.109
                Announcement bits (1): 0-mvpn global task
                AS path: I
                Communities: target:123:1 segmented-nh:10.255.162.109:0
                Import Accepted
                Localpref: 100
                Router ID: 10.255.162.109
                Primary Routing Table bgp.mvpn.0

3:10.255.162.109:100:32:172.16.1.2:32:192.0.2.3:10.255.162.109/240 (1 entry, 1 announced)
        *BGP    Preference: 170/-101
                PMSI: Flags 0x1: Label 0: Type INGRESS-REPLICATION 10.255.162.109
                Next hop type: Indirect, Next hop index: 0
                Address: 0xa5cddb0
                Next-hop reference count: 24
                Source: 10.255.162.109
                Protocol next hop: 10.255.162.109
                Indirect next hop: 0x0 - INH Session ID: 0x0
                State: <Secondary Active Int Ext>
                Local AS: 65550 Peer AS: 65550
                Age: 1:02:45    Metric2: 2
                Validation State: unverified
```

```
                  Task: BGP_65550.10.255.162.109
                  Announcement bits (1): 0-mvpn global task
                  AS path: I
                  Communities: target:123:1 segmented-nh:10.255.162.109:0
                  Import Accepted
                  Localpref: 100
                  Router ID: 10.255.162.109
                  Primary Routing Table bgp.mvpn.0
```

Meaning

The output shows the segmented type-3 traffic received from PE1 with the tunnel-type as IR.

Verification

IN THIS SECTION

Confirm that the configuration is working properly.

Verifying Segmented Type-3 Received from ABR2

Purpose

Display the segmented Type-3 received from ABR2 where the tunnel type is RSVP-TE.

Action

From operational mode, enter the `show route table vpn1.mvpn match-prefix 3:* detail` command.

```
user@ABR2> show route table vpn1.mvpn match-prefix 3:* detail

vpn1.mvpn.0: 22 destinations, 22 routes (22 active, 0 holddown, 0 hidden)
3:10.255.162.109:100:32:172.16.1.2:32:192.0.2.2:10.255.162.109/240 (1 entry, 1 announced)
```

```
        *BGP    Preference: 170/-101
                PMSI: Flags 0x1: Label 0: RSVP-TE: 
Session_13[10.255.162.100:0:6500:10.255.162.100]
                Next hop type: Indirect, Next hop index: 0
                Address: 0xa5bd650
                Next-hop reference count: 24
                Source: 10.255.162.100
                Protocol next hop: 10.255.162.109
                Indirect next hop: 0x0 - INH Session ID: 0x0
                State: <Secondary Active Int Ext>
                Local AS: 65550 Peer AS: 65550
                Age: 1:10:55    Metric2: 1
                Validation State: unverified
                Task: BGP_65550.10.255.162.100
                Announcement bits (1): 0-mvpn global task
                AS path: I (Originator)
                Cluster list:  0.0.0.1
                Originator ID: 10.255.162.109
                Communities: target:123:1 segmented-nh:10.255.162.100:0
                Import Accepted
                Localpref: 100
                Router ID: 10.255.162.100
                Primary Routing Table bgp.mvpn.0

3:10.255.162.109:100:32:172.16.1.2:32:192.0.2.1:10.255.162.109/240 (1 entry, 1 announced)
        *BGP    Preference: 170/-101
                PMSI: Flags 0x1: Label 0: RSVP-TE: 
Session_13[10.255.162.100:0:6504:10.255.162.100]
                Next hop type: Indirect, Next hop index: 0
                Address: 0xa5bd650
                Next-hop reference count: 24
                Source: 10.255.162.100
                Protocol next hop: 10.255.162.109
                Indirect next hop: 0x0 - INH Session ID: 0x0
                State: <Secondary Active Int Ext>
                Local AS: 65550 Peer AS: 65550
                Age: 1:10:25    Metric2: 1
                Validation State: unverified
                Task: BGP_65550.10.255.162.100
                Announcement bits (1): 0-mvpn global task
                AS path: I (Originator)
                Cluster list:  0.0.0.1
                Originator ID: 10.255.162.109
```

```
                Communities: target:123:1 segmented-nh:10.255.162.100:0
                Import Accepted
                Localpref: 100
                Router ID: 10.255.162.100
                Primary Routing Table bgp.mvpn.0

3:10.255.162.109:100:32:172.16.1.2:32:192.0.2.3:10.255.162.109/240 (1 entry, 1 announced)
        *BGP    Preference: 170/-101
                PMSI: Flags 0x1: Label 0: RSVP-TE:
Session_13[10.255.162.100:0:6502:10.255.162.100]
                Next hop type: Indirect, Next hop index: 0
                Address: 0xa5bd650
                Next-hop reference count: 24
                Source: 10.255.162.100
                Protocol next hop: 10.255.162.109
                Indirect next hop: 0x0 - INH Session ID: 0x0
                State: <Secondary Active Int Ext>
                Local AS: 65550 Peer AS: 65550
                Age: 1:10:55    Metric2: 1
                Validation State: unverified
                Task: BGP_65550.10.255.162.100
                Announcement bits (1): 0-mvpn global task
                AS path: I (Originator)
                Cluster list:  0.0.0.1
                Originator ID: 10.255.162.109
                Communities: target:123:1 segmented-nh:10.255.162.100:0
                Import Accepted
                Localpref: 100
                Router ID: 10.255.162.100
                Primary Routing Table bgp.mvpn.0
```

Meaning

The output displays the segmented Type-3 traffic received from ABR2 where the tunnel type is RSVP-TE.

Verifying Type-4 Received from Egress PE2 and PE4 and Locally Triggered Type-4 Toward Ingress ABR2

Purpose

Display the type-4 received from the egress PE2 and PE4 and locally triggered type-4 toward ingress ABR2.

Action

From operational mode, enter the `show route table vpn1.mvpn match-prefix 4:* detail` command.

```
user@ABR2> show route table vpn1.mvpn match-prefix 4:* detail

vpn1.mvpn.0: 22 destinations, 22 routes (22 active, 0 holddown, 0 hidden)
4:3:10.255.162.109:100:32:172.16.1.2:32:192.0.2.2:10.255.162.109:10.255.162.104/240 (1 entry, 1
announced)
        *BGP    Preference: 170/-101
                Next hop type: Indirect, Next hop index: 0
                Address: 0xa5d1720
                Next-hop reference count: 21
                Source: 10.255.162.104
                Protocol next hop: 10.255.162.104
                Indirect next hop: 0x0 - INH Session ID: 0x0
                State: <Secondary Active Int Ext>
                Local AS: 65550 Peer AS: 65550
                Age: 1:11:05    Metric2: 2
                Validation State: unverified
                Task: BGP_65550.10.255.162.104
                Announcement bits (1): 0-mvpn global task
                AS path: I
                Communities: target:10.255.162.117:0
                Import Accepted
                Localpref: 100
                Router ID: 10.255.162.104
                Primary Routing Table bgp.mvpn.0

4:3:10.255.162.109:100:32:172.16.1.2:32:192.0.2.2:10.255.162.109:10.255.162.117/240 (1 entry, 1
announced)
        *MVPN   Preference: 70
                Next hop type: Indirect, Next hop index: 0
                Address: 0xa5d31f0
```

```
                Next-hop reference count: 11
                Protocol next hop: 10.255.162.117
                Indirect next hop: 0x0 - INH Session ID: 0x0
                State: Active Int Ext
                Age: 1:11:04    Metric2: 1
                Validation State: unverified
                Task: mvpn global task
                Announcement bits (2): 0-mvpn global task 1-rt-export
                AS path: I
                Communities: target:10.255.162.100:0

4:3:10.255.162.109:100:32:172.16.1.2:32:192.0.2.2:10.255.162.109:198.51.100.17/240 (1 entry, 1 
announced)
        *BGP    Preference: 170/-101
                Next hop type: Indirect, Next hop index: 0
                Address: 0xa5cb0f0
                Next-hop reference count: 21
                Source: 198.51.100.17
                Protocol next hop: 198.51.100.17
                Indirect next hop: 0x0 - INH Session ID: 0x0
                State: <Secondary Active Int Ext>
                Local AS: 65550 Peer AS: 65550
                Age: 1:11:05    Metric2: 3
                Validation State: unverified
                Task: BGP_65550.198.51.100.17
                Announcement bits (1): 0-mvpn global task
                AS path: I
                Communities: target:10.255.162.117:0
                Import Accepted
                Localpref: 100
                Router ID: 198.51.100.17
                Primary Routing Table bgp.mvpn.0

4:3:10.255.162.109:100:32:172.16.1.2:32:192.0.2.1:10.255.162.109:10.255.162.104/240 (1 entry, 1 
announced)
        *BGP    Preference: 170/-101
                Next hop type: Indirect, Next hop index: 0
                Address: 0xa5d1720
                Next-hop reference count: 21
                Source: 10.255.162.104
                Protocol next hop: 10.255.162.104
                Indirect next hop: 0x0 - INH Session ID: 0x0
                State: <Secondary Active Int Ext>
```

```
                Local AS: 65550 Peer AS: 65550
                Age: 1:10:35    Metric2: 2
                Validation State: unverified
                Task: BGP_65550.10.255.162.104
                Announcement bits (1): 0-mvpn global task
                AS path: I
                Communities: target:10.255.162.117:0
                Import Accepted
                Localpref: 100
                Router ID: 10.255.162.104
                Primary Routing Table bgp.mvpn.0

4:3:10.255.162.109:100:32:172.16.1.2:32:192.0.2.1:10.255.162.109:10.255.162.117/240 (1 entry, 1
announced)
        *MVPN   Preference: 70
                Next hop type: Indirect, Next hop index: 0
                Address: 0xa5d31f0
                Next-hop reference count: 11
                Protocol next hop: 10.255.162.117
                Indirect next hop: 0x0 - INH Session ID: 0x0
                State: <Active Int Ext>
                Age: 1:10:35    Metric2: 1
                Validation State: unverified
                Task: mvpn global task
                Announcement bits (2): 0-mvpn global task 1-rt-export
                AS path: I
                Communities: target:10.255.162.100:0

4:3:10.255.162.109:100:32:172.16.1.2:32:192.0.2.1:10.255.162.109:198.51.100.17/240 (1 entry, 1
announced)
        *BGP    Preference: 170/-101
                Next hop type: Indirect, Next hop index: 0
                Address: 0xa5cb0f0
                Next-hop reference count: 21
                Source: 198.51.100.17
                Protocol next hop: 198.51.100.17
                Indirect next hop: 0x0 - INH Session ID: 0x0
                State: Secondary Active Int Ext
                Local AS: 65550 Peer AS: 65550
                Age: 1:10:35    Metric2: 3
                Validation State: unverified
                Task: BGP_65550.198.51.100.17
                Announcement bits (1): 0-mvpn global task
```

```
                AS path: I
                Communities: target:10.255.162.117:0
                Import Accepted
                Localpref: 100
                Router ID: 198.51.100.17
                Primary Routing Table bgp.mvpn.0

4:3:10.255.162.109:100:32:172.16.1.2:32:192.0.2.3:10.255.162.109:10.255.162.104/240 (1 entry, 1 
announced)
        *BGP    Preference: 170/-101
                Next hop type: Indirect, Next hop index: 0
                Address: 0xa5d1720
                Next-hop reference count: 21
                Source: 10.255.162.104
                Protocol next hop: 10.255.162.104
                Indirect next hop: 0x0 - INH Session ID: 0x0
                State: <Secondary Active Int Ext>
                Local AS: 65550 Peer AS: 65550
                Age: 1:11:04    Metric2: 2 
                Validation State: unverified 
                Task: BGP_65550.10.255.162.104
                Announcement bits (1): 0-mvpn global task 
                AS path: I
                Communities: target:10.255.162.117:0
                Import Accepted
                Localpref: 100
                Router ID: 10.255.162.104
                Primary Routing Table bgp.mvpn.0

4:3:10.255.162.109:100:32:172.16.1.2:32:192.0.2.3:10.255.162.109:10.255.162.117/240 (1 entry, 1 
announced)
        *MVPN   Preference: 70
                Next hop type: Indirect, Next hop index: 0
                Address: 0xa5d31f0
                Next-hop reference count: 11
                Protocol next hop: 10.255.162.117
                Indirect next hop: 0x0 - INH Session ID: 0x0
                State: <Active Int Ext>
                Age: 1:11:04    Metric2: 1 
                Validation State: unverified 
                Task: mvpn global task
                Announcement bits (2): 0-mvpn global task 1-rt-export 
                AS path: I
```

```
                Communities: target:10.255.162.100:0

4:3:10.255.162.109:100:32:172.16.1.2:32:192.0.2.3:10.255.162.109:198.51.100.17/240 (1 entry, 1
announced)
        *BGP    Preference: 170/-101
                Next hop type: Indirect, Next hop index: 0
                Address: 0xa5cb0f0
                Next-hop reference count: 21
                Source: 198.51.100.17
                Protocol next hop: 198.51.100.17
                Indirect next hop: 0x0 - INH Session ID: 0x0
                State: <Secondary Active Int Ext>
                Local AS: 65550 Peer AS: 65550
                Age: 1:11:04    Metric2: 3
                Validation State: unverified
                Task: BGP_65550.198.51.100.17
                Announcement bits (1): 0-mvpn global task
                AS path: I
                Communities: target:10.255.162.117:0
                Import Accepted
                Localpref: 100
                Router ID: 198.51.100.17
                Primary Routing Table bgp.mvpn.0
```

Meaning

The output shows that the configured tunnel type on the ABR2 is RSVP-TE. The RSVP tunnel from the ABR1 ends in ABR2 as the egress LSP, and the new LSP is triggered to egress PE2 and PE4.

Verifying the Statistics of MPLS LSP

Purpose

Display the statistics of MPLS LSP.

Action

From operational mode, run the `show mpls lsp statistics` command for Device ABR2.

```
user@ABR2> show mpls lsp statistics
```

```
Ingress LSP: 6 sessions
To              From            State     Packets            Bytes LSPname
10.255.162.104  10.255.162.117  Up              0                0
10.255.162.104:10.255.162.117:100:mv20:vpn1
10.255.162.104  10.255.162.117  Up              0                0
10.255.162.104:10.255.162.117:100:mv21:vpn1
10.255.162.104  10.255.162.117  Up              0                0
10.255.162.104:10.255.162.117:100:mv22:vpn1
198.51.100.17        10.255.162.117  Up              0                0
198.51.100.17:10.255.162.117:100:mv20:vpn1
198.51.100.17        10.255.162.117  Up              0                0
198.51.100.17:10.255.162.117:100:mv21:vpn1
198.51.100.17        10.255.162.117  Up              0                0
198.51.100.17:10.255.162.117:100:mv22:vpn1
Total 6 displayed, Up 6, Down 0

Egress LSP: 6 sessions
To              From            State     Packets            Bytes LSPname
10.255.162.117  10.255.162.100  Up             NA               NA
10.255.162.117:10.255.162.100:100:mv45:vpn1
10.255.162.117  10.255.162.100  Up             NA               NA
10.255.162.117:10.255.162.100:100:mv47:vpn1
10.255.162.117  10.255.162.100  Up             NA               NA
10.255.162.117:10.255.162.100:100:mv49:vpn1
10.255.162.117  10.255.162.104  Up             NA               NA PE2_1_to_ABR2
10.255.162.117  10.255.162.107  Up             NA               NA ABR3_to_ABR2
10.255.162.117  198.51.100.17        Up             NA               NA PE2_3_to_ABR2
Total 6 displayed, Up 6, Down 0
```

Verification

IN THIS SECTION

Confirm that the configuration is working properly.

Verifying Segmented Type-3 Received from ABR1 on ABR3

Purpose

Display the segmented Type-3 received from ABR1 on ABR3 where the tunnel type is RSVP-TE.

Action

From operational mode, run the `show route table vpn1.mvpn match-prefix 3:* detail` command for Device ABR3.

```
user@ABR3> show route table vpn1.mvpn match-prefix 3:* detail

vpn1.mvpn.0: 22 destinations, 22 routes (22 active, 0 holddown, 0 hidden)
3:10.255.162.109:100:32:172.16.1.2:32:192.0.2.2:10.255.162.109/240 (1 entry, 1 announced)
        *BGP    Preference: 170/-101
                PMSI: Flags 0x1: Label 0: RSVP-TE:
Session_13[10.255.162.100:0:6500:10.255.162.100]
                Next hop type: Indirect, Next hop index: 0
                Address: 0xa5bd650
                Next-hop reference count: 24
                Source: 10.255.162.100
                Protocol next hop: 10.255.162.109
                Indirect next hop: 0x0 - INH Session ID: 0x0
                State: <Secondary Active Int Ext>
                Local AS: 65550 Peer AS: 65550
                Age: 1:10:55    Metric2: 1
                Validation State: unverified
                Task: BGP_65550.10.255.162.100
                Announcement bits (1): 0-mvpn global task
                AS path: I (Originator)
                Cluster list:  0.0.0.1
                Originator ID: 10.255.162.109
                Communities: target:123:1 segmented-nh:10.255.162.100:0
                Import Accepted
                Localpref: 100
                Router ID: 10.255.162.100
                Primary Routing Table bgp.mvpn.0

3:10.255.162.109:100:32:172.16.1.2:32:192.0.2.1:10.255.162.109/240 (1 entry, 1 announced)
        *BGP    Preference: 170/-101
                PMSI: Flags 0x1: Label 0: RSVP-TE:
```

```
Session_13[10.255.162.100:0:6504:10.255.162.100]
                Next hop type: Indirect, Next hop index: 0
                Address: 0xa5bd650
                Next-hop reference count: 24
                Source: 10.255.162.100
                Protocol next hop: 10.255.162.109
                Indirect next hop: 0x0 - INH Session ID: 0x0
                State: <Secondary Active Int Ext>
                Local AS: 65550 Peer AS: 65550
                Age: 1:10:25    Metric2: 1
                Validation State: unverified
                Task: BGP_65550.10.255.162.100
                Announcement bits (1): 0-mvpn global task
                AS path: I (Originator)
                Cluster list:  0.0.0.1
                Originator ID: 10.255.162.109
                Communities: target:123:1 segmented-nh:10.255.162.100:0
                Import Accepted
                Localpref: 100
                Router ID: 10.255.162.100
                Primary Routing Table bgp.mvpn.0

3:10.255.162.109:100:32:172.16.1.2:32:192.0.2.3:10.255.162.109/240 (1 entry, 1 announced)
        *BGP    Preference: 170/-101
                PMSI: Flags 0x1: Label 0: RSVP-TE:
Session_13[10.255.162.100:0:6502:10.255.162.100]
                Next hop type: Indirect, Next hop index: 0
                Address: 0xa5bd650
                Next-hop reference count: 24
                Source: 10.255.162.100
                Protocol next hop: 10.255.162.109
                Indirect next hop: 0x0 - INH Session ID: 0x0
                State: <Secondary Active Int Ext>
                Local AS: 65550 Peer AS: 65550
                Age: 1:10:55    Metric2: 1
                Validation State: unverified
                Task: BGP_65550.10.255.162.100
                Announcement bits (1): 0-mvpn global task
                AS path: I (Originator)
                Cluster list:  0.0.0.1
                Originator ID: 10.255.162.109
                Communities: target:123:1 segmented-nh:10.255.162.100:0
                Import Accepted
```

```
                Localpref: 100
                Router ID: 10.255.162.100
                Primary Routing Table bgp.mvpn.0
```

Meaning

The output displays the segmented Type-3 traffic received from ABR1 where the tunnel type is RSVP-TE.

SEE ALSO

- Segmented Inter-Area Point-to-Multipoint Label-Switched Paths Overview | **629**
- Configuring Segmented Inter-Area P2MP LSP | **631**
- *all-regions*
- *inter-region*
- *inter-region-template*
- *inter-region-segmented*
- *region*
- *template*

Change History Table

Feature support is determined by the platform and release you are using. Use Feature Explorer to determine if a feature is supported on your platform.

Release	Description
11.1R2	Feature parity for the MVPN extranet functionality or overlapping MVPNs on the Junos Trio chipset is supported in Junos OS Releases 11.1R2, 11.2R2, and 11.4.

MVPN Route Distribution

IN THIS SECTION

This topic provides information and examples on configuring routing instances to support multicast in a Layer 3 VPN.

Configuring Routing Instances for an MBGP MVPN

To configure MBGP MVPNs, include the `mvpn` statement:

```
mvpn {
    mvpn-mode (rpt-spt | spt-only);
    receiver-site;
    route-target {
        export-target {
            target target-community;
            unicast;
        }
        import-target {
            target {
                target-value;
                receiver target-value;
                sender target-value;
            }
            unicast {
                receiver;
                sender;
            }
        }
    }
```

```
    sender-site;
    traceoptions {
        file filename <files number> <size size> <world-readable | no-world-readable>;
        flag flag <flag-modifier> <disable>;
    }
}
```

You can include this statement at the following hierarchy levels:

- `[edit routing-instances routing-instance-name protocols]`
- `[edit logical-systems logical-system-name routing-instances routing-instance-name protocols]`

By default an MBGP MVPN routing instance is associated with both the multicast sender and the receiver sites. If you configure the `receiver-site` option, the routing instance is associated with only multicast receiver sites. Configuring the `sender-site` option associates the routing instance with only multicast sender sites.

> **NOTE**: When you configure the routing instance for the MBGP MVPN, you must configure MPLS LSPs (either RSVP-signaled or LDP-signaled) between the PE routers of the routing instance to ensure VPN unicast connectivity. Point-to-multipoint LSPs are used for multicast data forwarding only.

Configuring Shared-Tree Data Distribution Across Provider Cores for Providers of MBGP MVPNs

For MBGP MVPNs (also referred to as next-generation Layer 3 multicast VPNs), the default mode of operation supports only intersite shortest-path trees (SPTs) for customer PIM (C-PIM) join messages. It does not support rendezvous-point trees (RPTs) for C-PIM join messages. The default mode of operation provides advantages, but it requires either that the customer rendezvous point (C-RP) be located on a PE router or that the Multicast Source Discovery Protocol (MSDP) be used between the C-RP and a PE router so that the PE router can learn about active sources advertised by other PE routers.

If the default mode is not suitable for your environment, you can configure RPT-SPT mode (also known as *shared-tree data distribution*), as documented in section 13 of the BGP-MVPN draft (draft-ietf-l3vpn-2547bis-mcast-bgp-00.txt). RPT-SPT mode supports the native PIM model of transmitting (*,G) messages from the receiver to the RP for intersite shared-tree join messages. This means that the type 6 (*,G) routes get transmitted from one PE router to another. In RPT-SPT mode, the shared-tree multicast routes are advertised from an egress PE router to the upstream router connected to the VPN site with the C-RP. The single-forwarder election is performed for the C-RP rather than for the source. The egress PE router takes the upstream hop to advertise the (*,G) and sends the type 6 route toward the upstream PE router. To send the data on the RPT, either inclusive or selective provider tunnels can be used. After the data starts flowing on the RPT, the last-hop router switches to SPT mode, unless you include the `spt-threshold infinity` statements in the configuration.

NOTE: The MVPN single-forwarder election follows the rule documented in section 9.1.1 of the BGP-MVPN draft (draft-ietf-l3vpn-2547bis-mcast-bgp-00.txt). The single-forwarder election winner is based on the following rules:

- If the active unicast route to the source is through the interface, then this route is used to determine the upstream multicast hop (UMH).
- If the active unicast route to the source is a VPN route, MVPN selects the UMH based on the highest IP address in the route import community for the VPN routes, and the local primary loopback address for local VRF routes.

The switch to SPT mode is performed by PIM and not by MVPN type 5 and type 6 routes. After the last-hop router switches to SPT mode, the SPT (S,G) join messages follow the same rules as the SPT-only default mode.

The advantage of RPT-SPT mode is that it provides a method for PE routers to discover sources in the multicast VPN when the C-RP is located on the customer site instead of on a PE router. Because the shared C-tree is established between VPN sites, there is no need to run MSDP between the C-RP and the PE routers. RPT-SPT mode also enables egress PE routers to switch to receiving data from the PE connected to the source after the source information is learned, instead of receiving data from the RP.

In Junos OS Release 15.1 and later, in RPT-SPT mode, PIM SSG Joins are created on the egress PE even if no directly-connected receivers are present.

CAUTION: When you configure RPT-SPT mode, receivers or sources directly attached to the PE router are not supported. As a workaround, place a CE router between any receiver or source and the PE router.

To configure RPT-SPT mode:

1. Enable shared-tree data distribution:

```
[edit routing-instances routing-instance-name protocols mvpn mvpn-mode]
user@router# set rpt-spt
```

2. Include the `rpt-spt` statement for all VRFs that make up the VPN.

Configuring SPT-Only Mode for Multiprotocol BGP-Based Multicast VPNs

For MBGP MVPNs (also referred to as next-generation Layer 3 multicast VPNs), the default mode of operation is shortest path tree only (SPT-only) mode. In SPT-only mode, the active multicast sources are learned through multicast VPN source-active routes. This mode of operation is described in section 14 of the BGP-MVPN draft (draft-ietf-l3vpn-2547bis-mcast-bgp-00.txt).

In contrast to SPT-only mode, rendezvous point tree (RPT)-SPT mode (also known as shared-tree data distribution) supports the native PIM model of transmitting (*,G) messages from the receiver to the RP for intersite shared-tree join messages.

In SPT-only mode, when a PE router receives a (*, C-G) join message, the router looks for an active source transmitting data to the customer group. If the PE router has a source-active route for the customer group, the router creates a source tree customer multicast route and sends the route to the PE router connected to the VPN site with the source. The source is determined by MVPN's single-forwarder election. When a receiver sends a (*,G) join message in a VPN site, the (*,G) join message only travels as far as the PE router. After the join message is converted to a type 7 multicast route, which is equivalent to a (S,G) join message, the route is installed with the no-advertise community setting.

NOTE: The MVPN single-forwarder election follows the rule documented in section 9.1.1 of the BGP-MVPN draft (draft-ietf-l3vpn-2547bis-mcast-bgp-00.txt). The single-forwarder election winner is based on the following rules:

- If the active unicast route to the source is through the interface, then this route is used to determine the upstream multicast hop (UMH).
- If the active unicast route to the source is a VPN route, MVPN selects the UMH based on the highest IP address in the route import community for the VPN routes, and the local primary loopback address for local VRF routes.

Single-forwarder election guarantees selection of a unique forwarder for a given customer source (C-S). The upstream PE router might differ for the source tree and the shared tree because the election is based on the customer source and C-RP, respectively. Although the single-forwarder election is sufficient for SPT-only mode, the alternative RPT-SPT mode involves procedures to prevent duplicate traffic from being sent on the shared tree and the source tree. These procedures might require administrator-configured parameters to reduce duplicate traffic and reduce null routes during RPT to SPT switch and the reverse.

In SPT-only mode, when a source is active, PIM creates a register state for the source both on the DR and on the C-RP (or on a PE router that is running Multicast Source Discovery Protocol [MSDP] between itself and the C-RP). After the register states are created, MVPN creates a source-active route. These type 5 source-active routes are installed on all PE routers. When the egress PE router with the (*,G) join message receives the source-active route, it has two routes that it can combine to produce the (S,G) multicast route. The type 7 route informs the PE router that a receiver is interested in group G. The source active route informs the PE router that a source S is transmitting data to group G. MVPN combines this information to produce a multicast join message and advertises this to the ingress PE router, as determined by the single-forwarder election.

For some service providers, the SPT-only implementation is not ideal because it creates a restriction on C-RP configuration. For a PE router to create customer multicast routes from (*, C-G) join messages, the router must learn about active sources through MVPN type 5 source-active routes. These source-active routes can be originated only by a PE router. This means that a PE router in the MVPN must learn about all PIM register messages sent to the RP, which is possible only in the following cases:

- The C-RP is colocated on one of the PEs in the MVPN.
- MSDP is run between the C-RP and the VRF instance on one of the PE routers in the MVPN.

If this restriction is not acceptable, providers can use RPT-SPT mode instead of the default SPT-only mode. However, because SPT-only mode does not transmit (*,G) routes between VPN sites, SPT-only mode has the following advantages over RPT-SPT mode:

- Simplified operations by exchanging and processing only source-tree customer multicast routes among PE routers
- Simplified operations by eliminating the need for the service provider to suppress MVPN transient duplicates during the switch from RPT to SPT
- Less control plane overhead in the service provider space by limiting the type of customer multicast routes exchanged, which results in more scalable deployments
- More stable traffic patterns in the backbone without the traffic shifts involved in the RPT-SPT mode
- Easier maintenance in the service provider space due to less state information

To configure SPT-only mode:

1. Explicitly configure SPT-only mode:

```
[edit routing-instances routing-instance-name protocols mvpn mvpn-mode]
user@router# set spt-only
```

2. Include the `spt-only` statement for all VRFs that make up the VPN.

Configuring Internet Multicast Using Ingress Replication Provider Tunnels

The routing instance type `mpls-internet-multicast` uses ingress replication provider tunnels to carry IP multicast data between routers through an MPLS cloud, enabling a faster path for multicast traffic between sender and receiver routers in large-scale implementations.

The `mpls-internet-multicast` routing instance is a non-forwarding instance used only for control plane procedures; it does not support any interface configurations. Only one `mpls-internet-multicast` routing instance can be defined for a logical system. All multicast and unicast routes used for Internet multicast are associated only with the master instance (inet.0), not with the routing instance.

Each router participating in Internet multicast must be configured with BGP MPLS-based Internet multicast for control plane procedures and with ingress replication for the data provider tunnel, which forms a full mesh of MPLS point-to-point LSPs. The ingress replication tunnel can be selective or inclusive, matching the configuration of the provider tunnel in the routing instance.

The topology consists of routers on the edge of the IP multicast domain that have a set of IP interfaces and a set of MPLS core-facing interfaces, see Figure 57 on page 716. Internet multicast traffic is carried between the IP routers, through the MPLS cloud, using ingress replication tunnels for the data plane and a full-mesh IGBP session for the control plane.

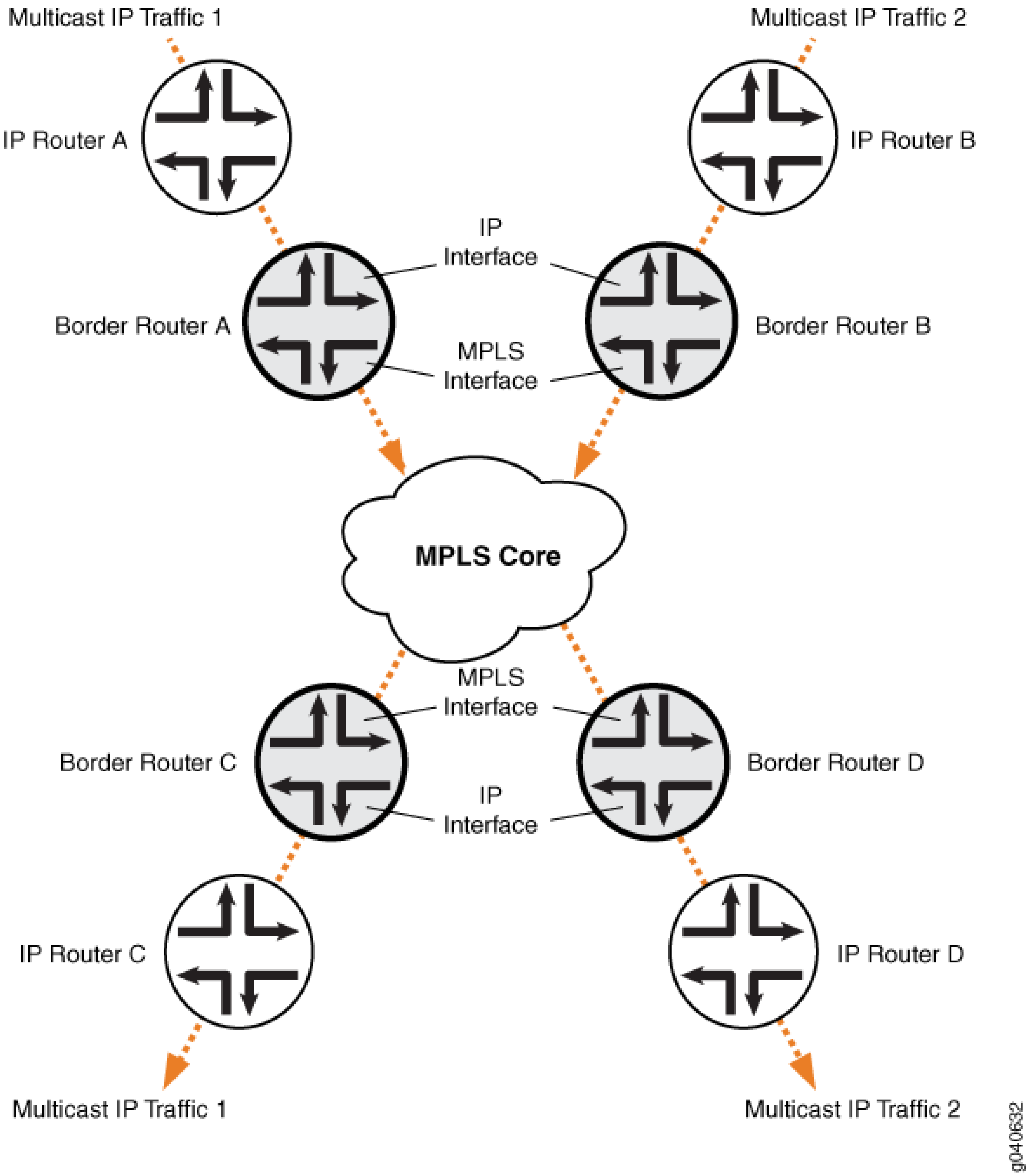
The `mpls-internet-multicast` routing instance type is configured for the default master instance on each router to support Internet multicast over MPLS. When using PIM as the multicast protocol, the `mpls-internet-multicast` configuration statement is also included at the `[edit protocols pim]` hierarchy level in the master instance. This creates a pseudo-interface that associates PIM with the `mpls-internet-multicast` routing instance.

When a new destination needs to be added to the ingress replication provider tunnel, the resulting behavior differs depending on how the ingress replication provider tunnel is configured:

- `create-new-ucast-tunnel`—When this statement is configured, a new unicast tunnel to the destination is created, and is deleted when the destination is no longer needed. Use this mode for RSVP LSPs using ingress replication.

- `label-switched-path-template (Multicast)`—When this statement is configured, an LSP template is used for the for the point-to-multipoint LSP for ingress replication.

Figure 57: Internet Multicast Topology



Example: Configure Internet Multicast Using Ingress Replication Tunnels

This example configures VPN-B with the instance type `mpls-internet-multicast`. This example also uses PIM for the multicast protocol.

1. Configure the routing instance type for VPN-B as `mpls-internet-multicast`:

```
user@host# set routing-instances VPN-B instance-type mpls-internet-multicast
```

2. Configure the ingress replication provider tunnel to create a new unicast tunnel each time an application requests to add a destination:

```
user@host# set routing-instances VPN-B provider-tunnel ingress-replication
create-new-ucast-tunnel
```

3. Configure the point-to-point LSP to use the default template settings.

```
user@host# set routing-instances VPN-B provider-tunnel ingress-replication
label-switched-path label-switched-path-template default-template
```

4. Configure the ingress replication provider tunnel to be selective:

```
user@host# set routing-instances VPN-B provider-tunnel selective
group 203.0.113.1/24 source 192.168.195.145/32 ingress-replication
label-switched-path
```

5. Configure MVPN protocol in the routing instance:

```
user@host# set routing-instances VPN-B protocols mvpn
```

6. Commit the configuration:

```
user@host# commit
```

7. Use show command to verify the instance has been created:

```
user@host# run show mvpn instance VPN-B
MVPN instance:
Legend for provider tunnel I-P-tnl -- inclusive provider tunnel S-P-tnl -- selective
provider tunnel
Legend for c-multicast routes properties (Pr)
DS -- derived from (*, c-g)          RM -- remote VPN route
Instance : VPN-B
   MVPN Mode : SPT-ONLY
```

```
   Provider tunnel: I-P-tnl:INGRESS-REPLICATION:MPLS Label 18:10.255.245.6
   Neighbor               I-P-tnl
   10.255.245.2           INGRESS-REPLICATION:MPLS Label 22:10.255.245.2
   10.255.245.7           INGRESS-REPLICATION:MPLS Label 19:10.255.245.7
   C-mcast IPv4 (S:G)     Ptnl                           St
   192.168.195.145/32:203.0.113.1/24 INGRESS-REPLICATION:MPLS Label
18:10.255.245.6         RM
```

8. Add the `mpls-internet-multicast` configuration statement under the `[edit protocols pim]` hierarchy level in the master instance:

```
user@host# set protocols pim mpls-internet-multicast
```

9. Commit the configuration:

```
user@host# commit
```

10. Use `show ingress-replication mvpn` command to verify configuration settings:

```
user@host# run show ingress-replication mvpn
Ingress Tunnel: mvpn:1l
  Application: MVPN
  Unicast tunnels
    Leaf Address       Tunnel-type       Mode       State
    10.255.245.2       P2P LSP           New        Up
    10.255.245.4       P2P LSP           New        Up
Ingress Tunnel: mvpn:2
  Application: MVPN
  Unicast tunnels
    Leaf Address       Tunnel-type       Mode       State
    10.255.245.2       P2P LSP           Existing   Up
```

11. Use this if you want to configure the ingress replication provider tunnel to be inclusive:

```
user@host# set routing-instances VPN-B provider-tunnel ingress-replication
create-new-ucast-tunnel
user@host# set routing-instances VPN-B provider-tunnel ingress-replication
label-switched-path label-switched-path-template default-template
```

12. Use show mvpn instance command to verify tunnel is inclusive:

```
user@host# run show mypn instance VPN-B
MVPN instance:
Legend for provider tunnel
I-P-tnl -- inclusive provider tunnel S-P-tnl -- selective provider tunnel

Legend for c-multicast routes properties (Pr)
DS -- derived from (*, c-g)          RM -- remote VPN route
Instance : VPN-A
  MVPN Mode : SPT-ONLY
  Provider tunnel: I-P-tnl:INGRESS-REPLICATION:MPLS Label 18:10.255.245.6
  Neighbor               I-P-tnl
  10.255.245.2           INGRESS-REPLICATION:MPLS Label 22:10.255.245.2
  10.255.245.7           INGRESS-REPLICATION:MPLS Label 19:10.255.245.7
  C-mcast IPv4 (S:G)     Ptnl                           St
  192.168.195.145/32:203.0.113.1/24 INGRESS-REPLICATION:MPLS Label 18:10.255.245.6
RM
```

SEE ALSO

create-new-ucast-tunnel

ingress-replication

mpls-internet-multicast

Provider Tunnel Selection In Ingress Replication

SUMMARY

Fine-tune the match criteria of unicast tunnels to RSVP/MLDP provider tunnels.

IN THIS SECTION

Configure Regular Expressions to Select RSVP Tunnels During Ingress Replication

Regular expression support for unicast tunnels enhances the match criteria of unicast tunnels to RSVP provider tunnels, enabling precise control over tunnel selection during ingress replication. You can fine-tune the selection criteria to determine which of the parallel provider tunnels towards an egress PE device is used for either a specific flow or for all flows, thus improving your multicast VPN (MVPN) deployments.

Figure 58: Selecting provider tunnels during Ingress Replication

For example, in the illustration above, there are two sets of RSVP tunnels with names "red" and "blue". You can further specify these names as a regular expression in the tunnel configuration (be it inclusive or selective) to match the corresponding set of unicast tunnels. When the regular expression of the unicast tunnel matches that of the RSVP tunnel LSP in the inet.3 table, that specific LSP is selected for ingress replication. This is applicable for both inclusive and selective PMSI. Use the `unicast-tunnel-name-regular-expression` configuration statement under the `routing-instances provider-tunnel ingress-replication` hierarchy.

- To configure **inclusive** PMSI:

```
set routing-instances routing-instance-name provider-tunnel ingress-replication unicast-
tunnel-name-regular-expression <reg-ex>
```

- To configure **selective** PMSI:

```
set routing-instances routing-instance-name provider-tunnel selective group 228.1.1.1/32
source 0.0.0.0/0 ingress-replication unicast-tunnel-name-regular-expression <reg-ex>
```

Configure a colored inet.3 table for Ingress Replication

By default, ingress replication uses the default inet.3 table to find the unicast tunnel to a tunnel leaf. Junos supports colored inet.3 tables for resolving BGP next hops, and they can also be used for ingress replication through unicast tunnels.

You can specify which RSVP tunnels are placed into specific colored inet.3 tables by using the configuration statement `transport-class` under the `routing-instances vrf provider-tunnel ingress-replication` hierarchy.

```
set routing-instances vrf provider-tunnel ingress-replication transport-class <color>
```

Subsequently configure the MVPN tunnel for ingress replication to specify which corresponding colored inet.3 table to use.

Fine-tuning RSVP Tunnel Selection by Combining Regular Expressions with Colored Tables

By combining regular expressions with colored inet.3 tables, you can further fine-tune RSVP tunnel selection during ingress replication. Only RSVP tunnel names that match a specific regular expression and belong to a specific colored inet.3 table are selected, providing granular control over which provider tunnels are used to route multicast traffic over the core network towards egress PEs.

For example, there are four LSPs to the same ingress replication tunnel leaf named red1, red2, blue1 and blue2. They're placed in red and blue colored inet.3 tables and the MVPN tunnel configuration for ingress replication is used to specify both a regular expression and the colored table to use. This ensures that only a tunnel in the specified table with the matching LSP name is used.

- To configure **inclusive** PMSI:

  ```
  set routing-instances vrf provider-tunnel ingress-replication transport-class <color> unicast-tunnel-name-regular-expression <reg-ex>
  ```

- To configure **selective** PMSI:

  ```
  set routing-instances vrf provider-tunnel selective group <group-address> source <source-address> ingress-replication transport-class <color> unicast-tunnel-name-regular-expression <reg-ex>
  ```

Configure a Root Address for MLDP Tunnels

IS-IS multi-instances can be used to create different topologies. By configuring the root address for MLDP tunnels, the lo.0 interface can be exported to different instances with different addresses. With this, egress PE devices can join tunnels in their corresponding topologies thereby achieving red/blue flow redundancy.

MVPN parses the configured root address and passes it to MLDP. In return, MLDP sends an MLDP FEC with different loopback addresses (as opposed to the default loopback) which is used by MVPN to advertise inclusive and selective tunnels via I-PMSI/S-PMSI routes. With different loopback addresses being exported into different instances, egress PE devices can join tunnels in its corresponding topologies.

Use the `root-address` configuration statement under the `routing-instances provider tunnel ldp-p2mp` hierarchy.

```
set routing-instances vrf provider-tunnel ldp-p2mp root-address <root-address>
```

SEE ALSO

> *unicast-tunnel-name-regular-expression*
>
> *transport-class*
>
> *root-address*

Controlling PIM Resources for Multicast VPNs Overview

IN THIS SECTION

A service provider network must protect itself from potential attacks from misconfigured or misbehaving customer edge (CE) devices and their associated VPN routing and forwarding (VRF) routing instances. Misbehaving CE devices can potentially advertise a large number of multicast routes toward a provider edge (PE) device, thereby consuming memory on the PE device and using other system resources in the network that are reserved for routes belonging to other VPNs.

To protect against potential misbehaving CE devices and VRF routing instances for specific multicast VPNs (MVPNs), you can control the following Protocol Independent Multicast (PIM) resources:

- Limit the number of accepted PIM join messages for any-source groups (*,G) and source-specific groups (S,G).

  Note how the device counts the PIM join messages:

  - Each (*,G) counts as one group toward the limit.
  - Each (S,G) counts as one group toward the limit.

- Limit the number of PIM register messages received for a specific VRF routing instance. Use this configuration if the device is configured as a rendezvous point (RP) or has the potential to become an RP. When a source in a multicast network becomes active, the source's designated router (DR) encapsulates multicast data packets into a PIM register message and sends them by means of unicast to the RP router.

  Note how the device counts PIM register messages:

  - Each unique (S,G) join received by the RP counts as one group toward the configured register messages limit.
  - Periodic register messages sent by the DR for existing or already known (S,G) entries do not count toward the configured register messages limit.
  - Register messages are accepted until either the PIM register limit or the PIM join limit (if configured) is exceeded. Once either limit isreached, any new requests are dropped.

- Limit the number of group-to-RP mappings allowed in a specific VRF routing instance. Use this configuration if the device is configured as an RP or has the potential to become an RP. This configuration can apply to devices configured for automatic RP announce and discovery (Auto-RP) or as a PIM bootstrap router. Every multicast device within a PIM domain must be able to map a particular multicast group address to the same RP. Both Auto-RP and the bootstrap router functionality are the mechanisms used to learn the set of group-to-RP mappings. Auto-RP is typically used in a PIM dense-mode deployment, and the bootstrap router is typically used in a PIM sparse-mode deployment.

  NOTE: The group-to-RP mappings limit does not apply to static RP or embedded RP configurations.

  Some important things to note about how the device counts group-to-RP mappings:

  - One group prefix mapped to five RPs counts as five group-to-RP mappings.
  - Five distinct group prefixes mapped to one RP count as five group-to-RP mappings.

Once the configured limits are reached, no new PIM join messages, PIM register messages, or group-to-RP mappings are accepted unless one of the following occurs:

- You clear the current PIM join states by using the `clear pim join` command. If you use this command on an RP configured for PIM register message limits, the register limit count is also restarted because the PIM join messages are unknown by the RP.

  NOTE: On the RP, you can also use the `clear pim register` command to clear all of the PIM registers. This command is useful if the current PIM register count is greater than the newly configured PIM register limit. After you clear the PIM registers, new PIM register messages are received up to the configured limit.

- The traffic responsible for the excess PIM join messages and PIM register messages stops and is no longer present.

- **CAUTION**: Never restart any of the software processes unless instructed to do so by a customer support engineer.

  You restart the PIM routing process on the device. This restart clears all of the configured limits but disrupts routing and therefore requires a maintenance window for the change.

System Log Messages for PIM Resources

You can optionally configure a system log warning threshold for each of the PIM resources. With this configuration, you can generate and review system log messages to detect if an excessive number of PIM join messages, PIM register messages, or group-to-RP mappings have been received on the device. The system log warning thresholds are configured per PIM resource and are a percentage of the configured maximum limits of the PIM join messages, PIM register messages, and group-to-RP mappings. You can further specify a log interval for each configured PIM resource, which is the amount of time (in seconds) between the log messages.

The log messages convey when the configured limits have been exceeded, when the configured warning thresholds have been exceeded, and when the configured limits drop below the configured warning threshold. Table 8 on page 724 describes the different types of PIM system messages that you might see depending on your system log warning and log interval configurations.

Table 8: PIM System Log Messages

System Log Message	Definition
RPD_PIM_SG_THRESHOLD_EXCEED	Records when the (S,G)/(*,G) routes exceed the configured warning threshold.

Table 8: PIM System Log Messages *(Continued)*

System Log Message	Definition
RPD_PIM_REG_THRESH_EXCEED	Records when the PIM registers exceed the configured warning threshold.
RPD_PIM_GRP_RP_MAP_THRES_EXCEED	Records when the group-to-RP mappings exceed the configured warning threshold.
RPD_PIM_SG_LIMIT_EXCEED	Records when the (S,G)/(*,G) routes exceed the configured limit, or when the configured log interval has been met and the routes exceed the configured limit.
RPD_PIM_REGISTER_LIMIT_EXCEED	Records when the PIM registers exceed the configured limit, or when the configured log interval has been met and the registers exceed the configured limit.
RPD_PIM_GRP_RP_MAP_LIMIT_EXCEED	Records when the group-to-RP mappings exceed the configured limit, or when the configured log interval has been met and the mapping exceeds the configured limit.
RPD_PIM_SG_LIMIT_BELOW	Records when the (S,G)/(*,G) routes drop below the configured limit and the configured log interval.
RPD_PIM_REGISTER_LIMIT_BELOW	Records when the PIM registers drop below the configured limit and the configured log interval.
RPD_PIM_GRP_RP_MAP_LIMIT_BELOW	Records when the group-to-RP mappings drop below the configured limit and the configured log interval.

Example: Configuring PIM State Limits

IN THIS SECTION

This example shows how to set limits on the Protocol Independent Multicast (PIM) state information so that a service provider network can protect itself from potential attacks from misconfigured or misbehaving customer edge (CE) devices and their associated VPN routing and forwarding (VRF) routing instances.

Requirements

No special configuration beyond device initialization is required before configuring this example.

Overview

In this example, a multiprotocol BGP-based multicast VPN (next-generation MBGP MVPN) is configured with limits on the PIM state resources.

The `sglimit maximum` statement sets a limit for the number of accepted (*,G) and (S,G) PIM join states received for the vpn-1 routing instance.

The `rp register-limit maximum` statement configures a limit for the number of PIM register messages received for the vpn-1 routing instance. You configure this statement on the rendezvuos point (RP) or on all the devices that might become the RP.

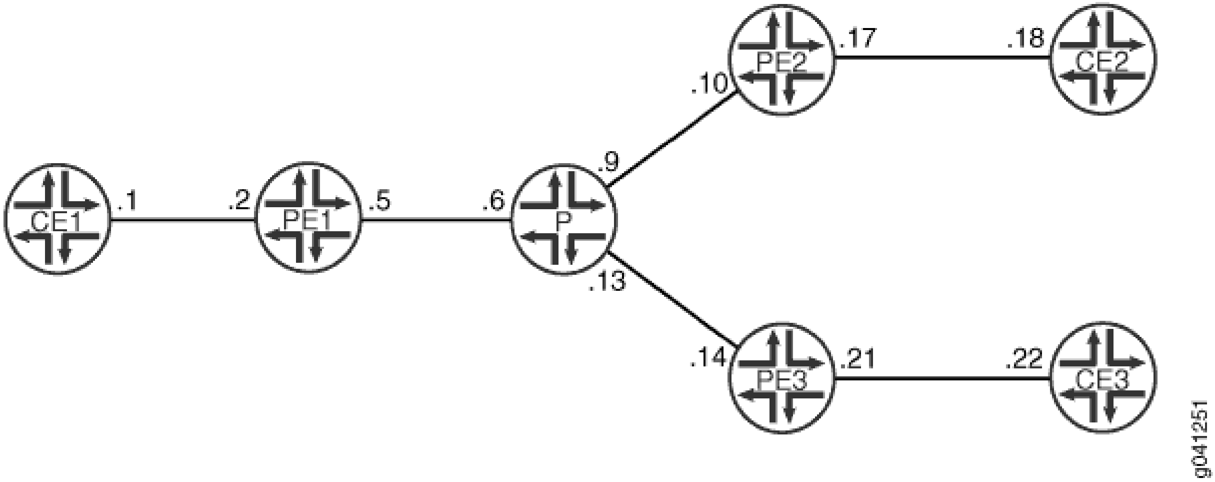
The `group-rp-mapping maximum` statement configures a limit for the number of group-to-RP mappings allowed in the vpn-1 routing instance.

For each configured PIM resource, the `threshold` statement sets a percentage of the maximum limit at which to start generating warning messages in the PIM log file.

For each configured PIM resource, the `log-interval` statement is an amount of time (in seconds) between system log message generation.

Figure 59 on page 727 shows the topology used in this example.

Figure 59: PIM State Limits Topology



"CLI Quick Configuration" on page 727 shows the configuration for all of the devices in Figure 59 on page 727. The section **Device PE1** below describes the steps for Device PE1.

Configuration

IN THIS SECTION

Procedure

CLI Quick Configuration

To quickly configure this example, copy the following commands, paste them into a text file, remove any line breaks, change any details necessary to match your network configuration, and then copy and paste the commands into the CLI at the `[edit]` hierarchy level.

Device CE1

```
set interfaces ge-1/2/0 unit 1 family inet address 10.1.1.1/30
set interfaces ge-1/2/0 unit 1 family mpls
set interfaces lo0 unit 1 family inet address 192.0.2.1/24
set protocols ospf area 0.0.0.0 interface lo0.1 passive
set protocols ospf area 0.0.0.0 interface ge-1/2/0.1
```

```
set protocols pim rp static address 203.0.113.1
set protocols pim interface all
set routing-options router-id 192.0.2.1
```

Device PE1

```
set interfaces ge-1/2/0 unit 2 family inet address 10.1.1.2/30
set interfaces ge-1/2/0 unit 2 family mpls
set interfaces ge-1/2/1 unit 5 family inet address 10.1.1.5/30
set interfaces ge-1/2/1 unit 5 family mpls
set interfaces vt-1/2/0 unit 2 family inet
set interfaces lo0 unit 2 family inet address 192.0.2.2/24
set interfaces lo0 unit 102 family inet address 203.0.113.1/24
set protocols mpls interface ge-1/2/1.5
set protocols bgp group ibgp type internal
set protocols bgp group ibgp local-address 192.0.2.2
set protocols bgp group ibgp family inet-vpn any
set protocols bgp group ibgp family inet-mvpn signaling
set protocols bgp group ibgp neighbor 192.0.2.4
set protocols bgp group ibgp neighbor 192.0.2.5
set protocols ospf area 0.0.0.0 interface lo0.2 passive
set protocols ospf area 0.0.0.0 interface ge-1/2/1.5
set protocols ldp interface ge-1/2/1.5
set protocols ldp p2mp
set policy-options policy-statement parent_vpn_routes from protocol bgp
set policy-options policy-statement parent_vpn_routes then accept
set routing-instances vpn-1 instance-type vrf
set routing-instances vpn-1 interface ge-1/2/0.2
set routing-instances vpn-1 interface vt-1/2/0.2
set routing-instances vpn-1 interface lo0.102
set routing-instances vpn-1 route-distinguisher 100:100
set routing-instances vpn-1 provider-tunnel ldp-p2mp
set routing-instances vpn-1 vrf-target target:1:1
set routing-instances vpn-1 protocols ospf export parent_vpn_routes
set routing-instances vpn-1 protocols ospf area 0.0.0.0 interface lo0.102 passive
set routing-instances vpn-1 protocols ospf area 0.0.0.0 interface ge-1/2/0.2
set routing-instances vpn-1 protocols pim sglimit family inet maximum 100
set routing-instances vpn-1 protocols pim sglimit family inet threshold 70
set routing-instances vpn-1 protocols pim sglimit family inet log-interval 10
set routing-instances vpn-1 protocols pim rp register-limit family inet maximum 100
set routing-instances vpn-1 protocols pim rp register-limit family inet threshold 80
set routing-instances vpn-1 protocols pim rp register-limit family inet log-interval 10
```

```
set routing-instances vpn-1 protocols pim rp group-rp-mapping family inet maximum 100
set routing-instances vpn-1 protocols pim rp group-rp-mapping family inet threshold 80
set routing-instances vpn-1 protocols pim rp group-rp-mapping family inet log-interval 10
set routing-instances vpn-1 protocols pim rp static address 203.0.113.1
set routing-instances vpn-1 protocols pim interface ge-1/2/0.2 mode sparse
set routing-instances vpn-1 protocols mvpn
set routing-options router-id 192.0.2.2
set routing-options autonomous-system 1001
```

Device P

```
set interfaces ge-1/2/0 unit 6 family inet address 10.1.1.6/30
set interfaces ge-1/2/0 unit 6 family mpls
set interfaces ge-1/2/1 unit 9 family inet address 10.1.1.9/30
set interfaces ge-1/2/1 unit 9 family mpls
set interfaces ge-1/2/2 unit 13 family inet address 10.1.1.13/30
set interfaces ge-1/2/2 unit 13 family mpls
set interfaces lo0 unit 3 family inet address 192.0.2.3/24
set protocols mpls interface ge-1/2/0.6
set protocols mpls interface ge-1/2/1.9
set protocols mpls interface ge-1/2/2.13
set protocols ospf area 0.0.0.0 interface lo0.3 passive
set protocols ospf area 0.0.0.0 interface ge-1/2/0.6
set protocols ospf area 0.0.0.0 interface ge-1/2/1.9
set protocols ospf area 0.0.0.0 interface ge-1/2/2.13
set protocols ldp interface ge-1/2/0.6
set protocols ldp interface ge-1/2/1.9
set protocols ldp interface ge-1/2/2.13
set protocols ldp p2mp
set routing-options router-id 192.0.2.3
```

Device PE2

```
set interfaces ge-1/2/0 unit 10 family inet address 10.1.1.10/30
set interfaces ge-1/2/0 unit 10 family mpls
set interfaces ge-1/2/1 unit 17 family inet address 10.1.1.17/30
set interfaces ge-1/2/1 unit 17 family mpls
set interfaces vt-1/2/0 unit 4 family inet
set interfaces lo0 unit 4 family inet address 192.0.2.4/24
set interfaces lo0 unit 104 family inet address 203.0.113.4/24
set protocols mpls interface ge-1/2/0.10
```

```
set protocols bgp group ibgp type internal
set protocols bgp group ibgp local-address 192.0.2.4
set protocols bgp group ibgp family inet-vpn any
set protocols bgp group ibgp family inet-mvpn signaling
set protocols bgp group ibgp neighbor 192.0.2.2
set protocols bgp group ibgp neighbor 192.0.2.5
set protocols ospf area 0.0.0.0 interface lo0.4 passive
set protocols ospf area 0.0.0.0 interface ge-1/2/0.10
set protocols ldp interface ge-1/2/0.10
set protocols ldp p2mp
set policy-options policy-statement parent_vpn_routes from protocol bgp
set policy-options policy-statement parent_vpn_routes then accept
set routing-instances vpn-1 instance-type vrf
set routing-instances vpn-1 interface vt-1/2/0.4
set routing-instances vpn-1 interface ge-1/2/1.17
set routing-instances vpn-1 interface lo0.104
set routing-instances vpn-1 route-distinguisher 100:100
set routing-instances vpn-1 vrf-target target:1:1
set routing-instances vpn-1 protocols ospf export parent_vpn_routes
set routing-instances vpn-1 protocols ospf area 0.0.0.0 interface lo0.104 passive
set routing-instances vpn-1 protocols ospf area 0.0.0.0 interface ge-1/2/1.17
set routing-instances vpn-1 protocols pim rp group-rp-mapping family inet maximum 100
set routing-instances vpn-1 protocols pim rp group-rp-mapping family inet threshold 80
set routing-instances vpn-1 protocols pim rp group-rp-mapping family inet log-interval 10
set routing-instances vpn-1 protocols pim rp static address 203.0.113.1
set routing-instances vpn-1 protocols pim interface ge-1/2/1.17 mode sparse
set routing-instances vpn-1 protocols mvpn
set routing-options router-id 192.0.2.4
set routing-options autonomous-system 1001
```

Device PE3

```
set interfaces ge-1/2/0 unit 14 family inet address 10.1.1.14/30
set interfaces ge-1/2/0 unit 14 family mpls
set interfaces ge-1/2/1 unit 21 family inet address 10.1.1.21/30
set interfaces ge-1/2/1 unit 21 family mpls
set interfaces vt-1/2/0 unit 5 family inet
set interfaces lo0 unit 5 family inet address 192.0.2.5/24
set interfaces lo0 unit 105 family inet address 203.0.113.5/24
set protocols mpls interface ge-1/2/0.14
set protocols bgp group ibgp type internal
set protocols bgp group ibgp local-address 192.0.2.5
```

```
set protocols bgp group ibgp family inet-vpn any
set protocols bgp group ibgp family inet-mvpn signaling
set protocols bgp group ibgp neighbor 192.0.2.2
set protocols bgp group ibgp neighbor 192.0.2.4
set protocols ospf area 0.0.0.0 interface lo0.5 passive
set protocols ospf area 0.0.0.0 interface ge-1/2/0.14
set protocols ldp interface ge-1/2/0.14
set protocols ldp p2mp
set policy-options policy-statement parent_vpn_routes from protocol bgp
set policy-options policy-statement parent_vpn_routes then accept
set routing-instances vpn-1 instance-type vrf
set routing-instances vpn-1 interface vt-1/2/0.5
set routing-instances vpn-1 interface ge-1/2/1.21
set routing-instances vpn-1 interface lo0.105
set routing-instances vpn-1 route-distinguisher 100:100
set routing-instances vpn-1 vrf-target target:1:1
set routing-instances vpn-1 protocols ospf export parent_vpn_routes
set routing-instances vpn-1 protocols ospf area 0.0.0.0 interface lo0.105 passive
set routing-instances vpn-1 protocols ospf area 0.0.0.0 interface ge-1/2/1.21
set routing-instances vpn-1 protocols pim rp static address 203.0.113.1
set routing-instances vpn-1 protocols pim interface ge-1/2/1.21 mode sparse
set routing-instances vpn-1 protocols mvpn
set routing-options router-id 192.0.2.5
set routing-options autonomous-system 1001
```

Device CE2

```
set interfaces ge-1/2/0 unit 18 family inet address 10.1.1.18/30
set interfaces ge-1/2/0 unit 18 family mpls
set interfaces lo0 unit 6 family inet address 192.0.2.6/24
set protocols sap listen 192.168.0.0
set protocols ospf area 0.0.0.0 interface lo0.6 passive
set protocols ospf area 0.0.0.0 interface ge-1/2/0.18
set protocols pim rp static address 203.0.113.1
set protocols pim interface all
set routing-options router-id 192.0.2.6
```

Device CE3

```
set interfaces ge-1/2/0 unit 22 family inet address 10.1.1.22/30
set interfaces ge-1/2/0 unit 22 family mpls
```

```
set interfaces lo0 unit 7 family inet address 192.0.2.7/24
set protocols ospf area 0.0.0.0 interface lo0.7 passive
set protocols ospf area 0.0.0.0 interface ge-1/2/0.22
set protocols pim rp static address 203.0.113.1
set protocols pim interface all
set routing-options router-id 192.0.2.7
```

Step-by-Step Procedure

The following example requires you to navigate various levels in the configuration hierarchy. For information about navigating the CLI, see *Using the CLI Editor in Configuration Mode* in the CLI User Guide.

To configure PIM state limits:

1. Configure the network interfaces.

   ```
   [edit interfaces]
   user@PE1# set ge-1/2/0 unit 2 family inet address 10.1.1.2/30
   user@PE1# set ge-1/2/0 unit 2 family mpls
   user@PE1# set ge-1/2/1 unit 5 family inet address 10.1.1.5/30
   user@PE1# set ge-1/2/1 unit 5 family mpls
   user@PE1# set vt-1/2/0 unit 2 family inet
   user@PE1# set lo0 unit 2 family inet address 192.0.2.2/24
   user@PE1# set lo0 unit 102 family inet address 203.0.113.1/24
   ```

2. Configure MPLS on the core-facing interface.

   ```
   [edit protocols mpls]
   user@PE1# set interface ge-1/2/1.5
   ```

3. Configure internal BGP (IBGP) on the main router.

   The IBGP neighbors are the other PE devices.

   ```
   [edit protocols bgp group ibgp]
   user@PE1# set type internal
   user@PE1# set local-address 192.0.2.2
   user@PE1# set family inet-vpn any
   user@PE1# set family inet-mvpn signaling
   ```

```
user@PE1# set neighbor 192.0.2.4
user@PE1# set neighbor 192.0.2.5
```

4. Configure OSPF on the main router.

```
[edit protocols ospf area 0.0.0.0]
user@PE1# set interface lo0.2 passive
user@PE1# set interface ge-1/2/1.5
```

5. Configure a signaling protocol (RSVP or LDP) on the main router.

```
[edit protocols ldp]
user@PE1# set interface ge-1/2/1.5
user@PE1# set p2mp
```

6. Configure the BGP export policy.

```
[edit policy-options policy-statement parent_vpn_routes]
user@PE1# set from protocol bgp
user@PE1# set then accept
```

7. Configure the routing instance.

   The customer-facing interfaces and the BGP export policy are referenced in the routing instance.

```
[edit routing-instances vpn-1]
user@PE1# set instance-type vrf
user@PE1# set interface ge-1/2/0.2
user@PE1# set interface vt-1/2/0.2
user@PE1# set interface lo0.102
user@PE1# set route-distinguisher 100:100
user@PE1# set provider-tunnel ldp-p2mp
user@PE1# set vrf-target target:1:1
user@PE1# set protocols ospf export parent_vpn_routes
user@PE1# set protocols ospf area 0.0.0.0 interface lo0.102 passive
user@PE1# set protocols ospf area 0.0.0.0 interface ge-1/2/0.2
user@PE1# set protocols pim rp static address 203.0.113.1
```

```
user@PE1# set protocols pim interface ge-1/2/0.2 mode sparse
user@PE1# set protocols mvpn
```

8. Configure the PIM state limits.

```
[edit routing-instances vpn-1 protocols pim]
user@PE1# set sglimit family inet maximum 100
user@PE1# set sglimit family inet threshold 70
user@PE1# set sglimit family inet log-interval 10
user@PE1# set rp register-limit family inet maximum 100
user@PE1# set rp register-limit family inet threshold 80
user@PE1# set rp register-limit family inet log-interval 10
user@PE1# set rp group-rp-mapping family inet maximum 100
user@PE1# set rp group-rp-mapping family inet threshold 80
user@PE1# set rp group-rp-mapping family inet log-interval 10
```

9. Configure the router ID and AS number.

```
[edit routing-options]
user@PE1# set router-id 192.0.2.2
user@PE1# set autonomous-system 1001
```

Results

From configuration mode, confirm your configuration by entering the `show interfaces`, `show protocols`, `show policy-options`, `show routing-instances`, and `show routing-options` commands. If the output does not display the intended configuration, repeat the configuration instructions in this example to correct it.

```
user@PE1# show interfaces
ge-1/2/0 {
    unit 2 {
        family inet {
            address 10.1.1.2/30;
        }
        family mpls;
    }
}
ge-1/2/1 {
    unit 5 {
```

```
        family inet {
            address 10.1.1.5/30;
        }
        family mpls;
    }
}
vt-1/2/0 {
    unit 2 {
        family inet;
    }
}
lo0 {
    unit 2 {
        family inet {
            address 192.0.2.2/24;
        }
    }
    unit 102 {
        family inet {
            address 203.0.113.1/24;
        }
    }
}
```

```
user@PE1# show protocols
mpls {
    interface ge-1/2/1.5;
}
bgp {
    group ibgp {
        type internal;
        local-address 192.0.2.2;
        family inet-vpn {
            any;
        }
        family inet-mvpn {
            signaling;
        }
        neighbor 192.0.2.4;
        neighbor 192.0.2.5;
    }
```

```
}
ospf {
    area 0.0.0.0 {
        interface lo0.2 {
            passive;
        }
        interface ge-1/2/1.5;
    }
}
ldp {
    interface ge-1/2/1.5;
    p2mp;
}
```

```
user@PE1# show policy-options
policy-statement parent_vpn_routes {
    from protocol bgp;
    then accept;
}
```

```
user@PE1# show routing-instances
vpn-1 {
    instance-type vrf;
    interface ge-1/2/0.2;
    interface vt-1/2/0.2;
    interface lo0.102;
    route-distinguisher 100:100;
    provider-tunnel {
        ldp-p2mp;
    }
    vrf-target target:1:1;
    protocols {
        ospf {
            export parent_vpn_routes;
            area 0.0.0.0 {
                interface lo0.102 {
                    passive;
                }
                interface ge-1/2/0.2;
            }
```

```
        }
        pim {
            sglimit {
                family inet {
                    maximum 100;
                    threshold 70;
                    log-interval 10;
                }
            }
            rp {
                register-limit {
                    family inet {
                        maximum 100;
                        threshold 80;
                        log-interval 10;
                    }
                }
                group-rp-mapping {
                    family inet {
                        maximum 100;
                        threshold 80;
                        log-interval 10;
                    }
                }
                static {
                    address 203.0.113.1;
                }
            }
            interface ge-1/2/0.2 {
                mode sparse;
            }
        }
        mvpn;
    }
}
```

```
user@PE1# show routing-options
router-id 192.0.2.2;
autonomous-system 1001;
```

If you are done configuring the device, enter **commit** from configuration mode.

Verification

IN THIS SECTION

Confirm that the configuration is working properly.

Monitoring the PIM State Information

Purpose

Verify that the counters are set as expected and are not exceeding the configured limits.

Action

From operational mode, enter the `show pim statistics` command.

```
user@PE1> show pim statistics instance vpn-1
PIM Message type        Received       Sent  Rx errors
V2 Hello                     393        390          0
...
V4 (S,G) Maximum                            100
V4 (S,G) Accepted                             0
V4 (S,G) Threshold                           70
V4 (S,G) Log Interval                        10
V4 (grp-prefix, RP) Maximum                 100
V4 (grp-prefix, RP) Accepted                  0
V4 (grp-prefix, RP) Threshold                80
V4 (grp-prefix, RP) Log Interval             10
V4 Register Maximum                         100
V4 Register Accepted                          0
V4 Register Threshold                        80
V4 Register Log Interval                     10
```

Meaning

The V4 (S,G) Maximum field shows the maximum number of (S,G) IPv4 multicast routes accepted for the VPN routing instance. If this number is met, additional (S,G) entries are not accepted.

The V4 (S,G) Accepted field shows the number of accepted (S,G) IPv4 multicast routes.

The V4 (S,G) Threshold field shows the threshold at which a warning message is logged (percentage of the maximum number of (S,G) IPv4 multicast routes accepted by the device).

The V4 (S,G) Log Interval field shows the time (in seconds) between consecutive log messages.

The V4 (grp-prefix, RP) Maximum field shows the maximum number of group-to-rendezvous point (RP) IPv4 multicast mappings accepted for the VRF routing instance. If this number is met, additional mappings are not accepted.

The V4 (grp-prefix, RP) Accepted field shows the number of accepted group-to-RP IPv4 multicast mappings.

The V4 (grp-prefix, RP) Threshold field shows the threshold at which a warning message is logged (percentage of the maximum number of group-to-RP IPv4 multicast mappings accepted by the device).

The V4 (grp-prefix, RP) Log Interval field shows the time (in seconds) between consecutive log messages.

The V4 Register Maximum field shows the maximum number of IPv4 PIM registers accepted for the VRF routing instance. If this number is met, additional PIM registers are not accepted. You configure the register limits on the RP.

The V4 Register Accepted field shows the number of accepted IPv4 PIM registers.

The V4 Register Threshold field shows the threshold at which a warning message is logged (percentage of the maximum number of IPv4 PIM registers accepted by the device).

The V4 Register Log Interval field shows the time (in seconds) between consecutive log messages.

Understanding Wildcards to Configure Selective Point-to-Multipoint LSPs for an MBGP MVPN

IN THIS SECTION

Selective LSPs are also referred to as selective provider tunnels. Selective provider tunnels carry traffic from some multicast groups in a VPN and extend only to the PE routers that have receivers for these groups. You can configure a selective provider tunnel for group prefixes and source prefixes, or you can use wildcards for the group and source, as described in the Internet draft draft-rekhter-mvpn-wildcard-spmsi-01.txt, *Use of Wildcard in S-PMSI Auto-Discovery Routes*.

The following sections describe the scenarios and special considerations when you use wildcards for selective provider tunnels.

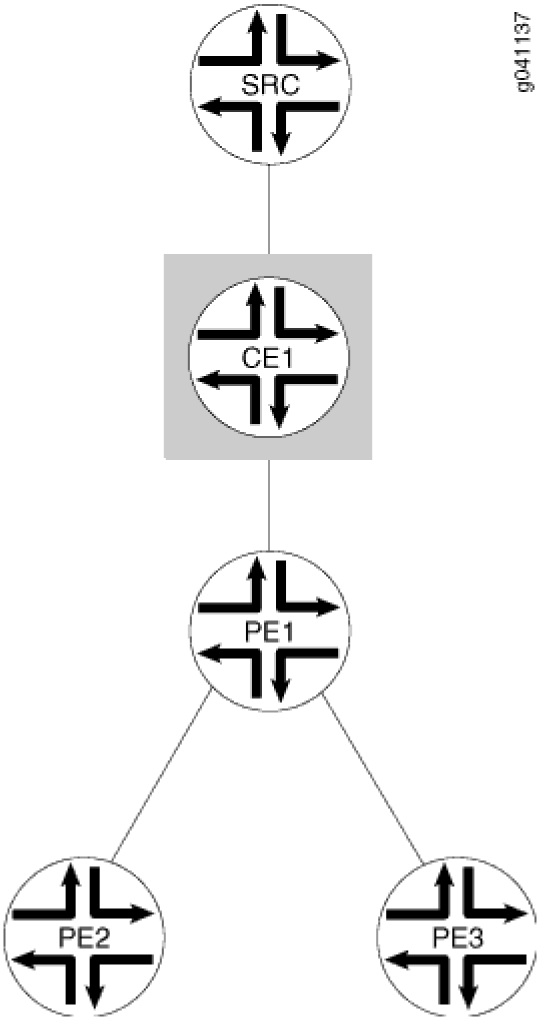
About S-PMSI

The provider multicast service interface (PMSI) is a BGP tunnel attribute that contains the tunnel ID used by the PE router for transmitting traffic through the core of the provider network. A selective PMSI (S-PMSI) autodiscovery route advertises binding of a given MVPN customer multicast flow to a particular provider tunnel. The S-PMSI autodiscovery route advertised by the ingress PE router contains /32 IPv4 or /128 IPv6 addresses for the customer source and the customer group derived from the source-tree customer multicast route.

Figure 60 on page 741 shows a simple MVPN topology. The ingress router, PE1, originates the S-PMSI autodiscovery route. The egress routers, PE2 and PE3, have join state as a result of receiving join messages from CE devices that are not shown in the topology. In response to the S-PMSI autodiscovery route advertisement sent by PE1, PE2, and PE3, elect whether or not to join the tunnel based on the join state. The selective provider tunnel configuration is configured in a VRF instance on PE1.

NOTE: The MVPN mode configuration (RPT-SPT or SPT-only) is configured on all three PE routers for all VRFs that make up the VPN. If you omit the MVPN mode configuration, the default mode is SPT-only.

Figure 60: Simple MVPN Topology



Scenarios for Using Wildcard S-PMSI

A wildcard S-PMSI has the source or the group (or both the source and the group) field set to the wildcard value of 0.0.0.0/0 and advertises binding of multiple customer multicast flows to a single provider tunnel in a single S-PMSI autodiscovery route.

The scenarios under which you might configure a wildcard S-PMSI are as follows:

- When the customer multicast flows are PIM-SM in ASM-mode flows. In this case, a PE router connected to an MVPN customer's site that contains the customer's RP (C-RP) could bind all the customer multicast flows traveling along a customer's RPT tree to a single provider tunnel.
- When a PE router is connected to an MVPN customer’s site that contains multiple sources, all sending to the same group.

- When the customer multicast flows are PIM-bidirectional flows. In this case, a PE router could bind to a single provider tunnel all the customer multicast flows for the same group that have been originated within the sites of a given MVPN connected to that PE, and advertise such binding in a single S-PMSI autodiscovery route.
- When the customer multicast flows are PIM-SM in SSM-mode flows. In this case, a PE router could bind to a single provider tunnel all the customer multicast flows coming from a given source located in a site connected to that PE router.
- When you want to carry in the provider tunnel all the customer multicast flows originated within the sites of a given MVPN connected to a given PE router.

Types of Wildcard S-PMSI

The following types of wildcard S-PMSI are supported:

- A (*,G) S-PMSI matches all customer multicast routes that have the group address. The customer source address in the customer multicast route can be any address, including 0.0.0.0/0 for shared-tree customer multicast routes. A (*, C-G) S-PMSI autodiscovery route is advertised with the source field set to 0 and the source address length set to 0. The multicast group address for the S-PMSI autodiscovery route is derived from the customer multicast joins.
- A (*,*) S-PMSI matches all customer multicast routes. Any customer source address and any customer group address in a customer multicast route can be bound to the (*,*) S-PMSI. The S-PMSI autodiscovery route is advertised with the source address and length set to 0 and the group address and length set 0. The remaining fields in the S-PMSI autodiscovery route follow the same rule as (C-S, C-G) S-PMSI, as described in section 12.1 of the BGP-MVPN draft (draft-ietf-l3vpn-2547bis-mcast-bgp-00.txt).

Differences Between Wildcard S-PMSI and (S,G) S-PMSI

For dynamic provider tunnels, each customer multicast stream is bound to a separate provider tunnel, and each tunnel is advertised by a separate S-PMSI autodiscovery route. For static LSPs, multiple customer multicast flows are bound to a single provider tunnel by having multiple S-PMSI autodiscovery routes advertise the same provider tunnel.

When you configure a wildcard (*,G) or (*,*) S-PMSI, one or more matching customer multicast routes share a single S-PMSI. All customer multicast routes that have a matching source and group address are bound to the same (*,G) or (*,*) S-PMSI and share the same tunnel. The (*,G) or (*,*) S-PMSI is established when the first matching remote customer multicast join message is received in the ingress PE router, and deleted when the last remote customer multicast join is withdrawn from the ingress PE router. Sharing a single S-PMSI autodiscovery route improves control plane scalability.

Wildcard (*,*) S-PMSI and PIM Dense Mode

For (S,G) and (*,G) S-PMSI autodiscovery routes in PIM dense mode (PIM-DM), all downstream PE routers receive PIM-DM traffic. If a downstream PE router does not have receivers that are interested in the group address, the PE router instantiates prune state and stops receiving traffic from the tunnel.

Now consider what happens for (*,*) S-PMSI autodiscovery routes. If the PIM-DM traffic is not bound by a longer matching (S,G) or (*,G) S-PMSI, it is bound to the (*,*) S-PMSI. As is always true for dense mode, PIM-DM traffic is flooded to downstream PE routers over the provider tunnel regardless of the customer multicast join state. Because there is no group information in the (*,*) S-PMSI autodiscovery route, egress PE routers join a (*,*) S-PMSI tunnel if there is any configuration on the egress PE router indicating interest in PIM-DM traffic.

Interest in PIM-DM traffic is indicated if the egress PE router has one of the following configurations in the VRF instance that corresponds to the instance that imports the S-PMSI autodiscovery route:

- At least one interface is configured in dense mode at the `[edit routing-instances` *`instance-name`* `protocols pim interface]` hierarchy level.
- At least one group is configured as a dense-mode group at the `[edit routing-instances` *`instance-name`* `protocols pim dense-groups` *`group-address`*`]` hierarchy level.

Wildcard (*,*) S-PMSI and PIM-BSR

For (S,G) and (*,G) S-PMSI autodiscovery routes in PIM bootstrap router (PIM-BSR) mode, an ingress PE router floods the PIM bootstrap message (BSM) packets over the provider tunnel to all egress PE routers. An egress PE router does not join the tunnel unless the message has the ALL-PIM-ROUTERS group. If the message has this group, the egress PE router joins the tunnel, regardless of the join state. The group field in the message determines the presence or absence of the ALL-PIM-ROUTERS address.

Now consider what would happen for (*,*) S-PMSI autodiscovery routes used with PIM-BSR mode. If the PIM BSM packets are not bound by a longer matching (S,G) or (*,G) S-PMSI, they are bound to the (*,*) S-PMSI. As is always true for PIM-BSR, BSM packets are flooded to downstream PE routers over the provider tunnel to the ALL-PIM-ROUTERS destination group. Because there is no group information in the (*,*) S-PMSI autodiscovery route, egress PE routers always join a (*,*) S-PMSI tunnel. Unlike PIM-DM, the egress PE routers might have no configuration suggesting use of PIM-BSR as the RP discovery mechanism in the VRF instance. To prevent all egress PE routers from always joining the (*,*) S-PMSI tunnel, the (*,*) wildcard group configuration must be ignored.

This means that if you configure PIM-BSR, a wildcard-group S-PMSI can be configured for all other group addresses. The (*,*) S-PMSI is not used for PIM-BSR traffic. Either a matching (*,G) or (S,G) S-PMSI (where the group address is the ALL-PIM-ROUTERS group) or an inclusive provider tunnel is needed to transmit data over the provider core. For PIM-BSR, the longest-match lookup is (S,G), (*,G), and the inclusive provider tunnel, in that order. If you do not configure an inclusive tunnel for the routing instance, you must configure a (*,G) or (S,G) selective tunnel. Otherwise, the data is dropped. This is because PIM-BSR functions like PIM-DM, in that traffic is flooded to downstream PE routers over the provider tunnel regardless of the customer multicast join state. However, unlike PIM-DM, the egress PE routers might have no configuration to indicate interest or noninterest in PIM-BSR traffic.

Wildcard Source and the 0.0.0.0/0 Source Prefix

You can configure a 0.0.0.0/0 source prefix and a wildcard source under the same group prefix in a selective provider tunnel. For example, the configuration might look as follows:

```
routing-instances {
    vpna {
        provider-tunnel {
            selective {
                group 203.0.113.0/24 {
                    source 0.0.0.0/0 {
                        rsvp-te {
                            label-switched-path-template {
                                sptnl3;
                            }
                        }
                    }
                    wildcard-source {
                        rsvp-te {
                            label-switched-path-template {
                                sptnl2;
                            }
                            static-lsp point-to-multipoint-lsp-name;
                        }
                        threshold-rate kbps;
                    }
                }
            }
        }
    }
}
```

The functions of the `source 0.0.0.0/0` and `wildcard-source` configuration statements are different. The 0.0.0.0/0 source prefix only matches (C-S, C-G) customer multicast join messages and triggers (C-S, C-G) S-PMSI autodiscovery routes derived from the customer multicast address. Because all (C-S, C-G) join messages are matched by the 0.0.0.0/0 source prefix in the matching group, the wildcard source S-PMSI is used only for (*,C-G) customer multicast join messages. In the absence of a configured 0.0.0.0/0 source prefix, the wildcard source matches (C-S, C-G) and (*,C-G) customer multicast join messages. In the example, a join message for (10.0.1.0/24, 203.0.113.0/24) is bound to `sptnl3`. A join message for (*, 203.0.113.0/24) is bound to `sptnl2`.

Configuring a Selective Provider Tunnel Using Wildcards

When you configure a selective provider tunnel for MBGP MVPNs (also referred to as next-generation Layer 3 multicast VPNs), you can use wildcards for the multicast group and source address prefixes. Using wildcards enables a PE router to use a single route to advertise the binding of multiple multicast streams of a given MVPN customer to a single provider's tunnel, as described in https://tools.ietf.org/html/draft-rekhter-mvpn-wildcard-spmsi-00 .

Sharing a single route improves control plane scalability because it reduces the number of S-PMSI autodiscovery routes.

To configure a selective provider tunnel using wildcards:

1. Configure a wildcard group matching any group IPv4 address and a wildcard source for (*,*) join messages.

   ```
   [edit routing-instances vpna provider-tunnel selective]
   user@router# set wildcard-group-inet wildcard-source
   ```

2. Configure a wildcard group matching any group IPv6 address and a wildcard source for (*,*) join messages.

   ```
   [edit routing-instances vpna provider-tunnel selective]
   user@router# set wildcard-group-inet6 wildcard-source
   ```

3. Configure an IP prefix of a multicast group and a wildcard source for (*,G) join messages.

   ```
   [edit routing-instances vpna provider-tunnel selective]
   user@router# set group 203.0.113/24 wildcard-source
   ```

4. Map the IPv4 join messages to a selective provider tunnel.

   ```
   [edit routing-instances vpna provider-tunnel selective wildcard-group-inet wildcard-source]
   user@router# set rsvp-te (Routing Instances Provider Tunnel Selective) label-switched-path-
   template provider-tunnel1
   ```

5. Map the IPv6 join messages to a selective provider tunnel.

   ```
   [edit routing-instances vpna provider-tunnel selective wildcard-group-inet6 wildcard-source]
   user@router# set rsvp-te (Routing Instances Provider Tunnel Selective) label-switched-path-
   template provider-tunnel2
   ```

6. Map the (*,203.0.113/24) join messages to a selective provider tunnel.

   ```
   [edit routing-instances vpna provider-tunnel selective group 203.0.113/24 wildcard-source]
   user@router# set rsvp-te (Routing Instances Provider Tunnel Selective) label-switched-path-
   template provider-tunnel3
   ```

Example: Configuring Selective Provider Tunnels Using Wildcards

With the (*,G) and (*,*) S-PMSI, a customer multicast join message can match more than one S-PMSI. In this case, a customer multicast join message is bound to the longest matching S-PMSI. The longest match is a (S,G) S-PMSI, followed by a (*,G) S-PMSI and a (*,*) S-PMSI, in that order.

Consider the following configuration:

```
routing-instances {
    vpna {
        provider-tunnel {
            selective {
                wildcard-group-inet {
                    wildcard-source {
                        rsvp-te {
                            label-switched-path-template {
                                sptnl1;
                            }
                        }
                    }
                }
                group 203.0.113.0/24 {
                    wildcard-source {
                        rsvp-te {
                            label-switched-path-template {
                                sptnl2;
                            }
                        }
                    }
```

```
                    source 10.1.1/24 {
                        rsvp-te {
                            label-switched-path-template {
                                sptnl3;
                            }
                        }
                    }
                }
            }
        }
    }
}
```

For this configuration, the longest-match rule works as follows:

- A customer multicast (10.1.1.1, 203.0.113.1) join message is bound to the sptnl3 S-PMSI autodiscovery route.
- A customer multicast (10.2.1.1, 203.0.113.1) join message is bound to the sptnl2 S-PMSI autodiscovery route.
- A customer multicast (10.1.1.1, 203.1.113.1) join message is bound to the sptnl1 S-PMSI autodiscovery route.

When more than one customer multicast route is bound to the same wildcard S-PMSI, only one S-PMSI autodiscovery route is created. An egress PE router always uses the same matching rules as the ingress PE router that advertises the S-PMSI autodiscovery route. This ensures consistent customer multicast mapping on the ingress and the egress PE routers.

Configuring NLRI Parameters for an MBGP MVPN

To enable VPN signaling where multiprotocol BGP carries multicast VPN NLRI for the IPv4 address family, include the `family inet-mvpn` statement:

```
inet-mvpn {
    signaling {
        accepted-prefix-limit {
            maximum number;
            teardown percentage {
                idle-timeout (forever | minutes);
            }
        }
        loops number;
        prefix-limit {
```

```
                maximum number;
                teardown percentage {
                    idle-timeout (forever | minutes);
                }
            }
        }
    }
```

To enable VPN signaling where multiprotocol BGP carries multicast VPN NLRI for the IPv6 address family, include the `family inet6-mvpn` statement:

```
inet6-mvpn {
    signaling {
        accepted-prefix-limit {
            maximum number;
            teardown percentage {
                idle-timeout (forever | minutes);
            }
        }
        loops number
        prefix-limit {
            maximum number;
            teardown percentage {
                idle-timeout (forever | minutes);
            }
        }
    }
}
```

Resiliency in Multicast L3 VPNs with Redundant Virtual Tunnels

IN THIS SECTION

Redundant Virtual Tunnels Providing Resiliency in Delivering Multicast Traffic Overview

In multicast Layer 3 VPNs, virtual tunnel (VT) interfaces are needed to facilitate virtual routing and forwarding (VRF) table lookup based on MPLS labels.

Junos OS supports *redundant VTs* at the Packet Forwarding Engine level to improve resiliency in delivering multicast traffic.

NOTE: Redundant VTs are supported only on MX Series routers with MPCs.

To create redundant VTs at the Packet Forwarding Engine level, you add member VTs (vt- interfaces) to a parent VT (rvt interface).

Configuring redundant VT interfaces on MX Series routers with MPCs has the following characteristics:

- When creating a redundant VT, you can add unconfigured VTs as members of the redundant VT. You configure the redundant VT the same way you configure a regular VT, and the member VTs inherit the configuration of the parent redundant VT.
- When a VT with an existing configuration joins a redundant VT, you must configure the redundant VT with the settings from the existing configuration.
- You can create up to 16 redundant VTs.
- You can add up to 32 VTs as members of the redundant VTs.

  NOTE: The actual number of VTs you can create is determined by the type of chassis and the number of line cards you have.

- When you add more than two VTs to a redundant VT, the members are in active mode by default, and traffic through the redundant VT is load balanced across all members of the redundant VT.
- When you add only two members, you can configure the members in one of two ways:
  - Both members in active mode
  - One member in active mode and the other in backup mode

BEST PRACTICE: To set one member to active mode and the other to backup mode, we recommend that the members be hosted on different MPCs. That way, if an MPC fails, it does not bring down the entire rvt interface.

- Class of service and firewall configurations involving VT interfaces work the same on redundant VT interfaces.

NOTE: Redundant VTs help to provide resiliency and improve uptime in your network when delivering multicast traffic. When `enhanced-ip` is enabled at the `[edit chassis network-services]` hierarchy level, failure of a VT that is a member of a redundant VT can typically be detected and fail over to another member VT provided within 50 milliseconds.

Configuring Redundant Virtual Tunnels to Provide Resiliency in Delivering Multicast Traffic

In multicast Layer 3 VPNs, virtual tunnel (VT) interfaces are needed to facilitate virtual routing and forwarding (VRF) table lookup based on MPLS labels.

Junos OS supports *redundant VTs* at the Packet Forwarding Engine level to improve resiliency in delivering multicast traffic.

NOTE: Redundant VTs are supported only on MX Series routers with MPCs.

To create redundant VTs at the Packet Forwarding Engine level, add member VTs (vt- interfaces) to a redundant VT (rvt interface), and add that redundant VT interface to a vrf routing instance.

To configure a redundant VT:

1. Enable tunnel services on the MX Series router.

   ```
   [edit chassis]
   user@host# set fpc slot-number pic number tunnel-services bandwidth (1g | 10g | 20g | 40g)
   ```

   For example:

   ```
   [edit chassis]
   user@host# set fpc 3 pic 0 tunnel-services bandwidth 1g
   user@host# set fpc 3 pic 2 tunnel-services bandwidth 10g
   ```

See Tunnel Interface Configuration on MX Series Routers Overview for more information.

2. Create the redundant VT interface type.

```
[edit chassis]
user@host# set redundancy-group interface-type redundant-virtual-tunnel device-count count
```

For example:

```
[edit chassis]
user@host# set redundancy-group interface-type redundant-virtual-tunnel device-count 4
```

3. Enable `enhanced-ip` under network services.

```
[edit chassis network-services]
user@host# set enhanced-ip
```

4. Create the redundant VT interface.

```
[edit interfaces interface-name]
user@host# set redundancy-group member-interface interface-name
```

For example:

```
[edit interfaces rvt0]
user@host# set redundancy-group member-interface vt-3/0/10
user@host# set redundancy-group member-interface vt-3/2/10
```

See *redundancy-group (Interfaces)* for more information.

> **NOTE**: You configure the remaining parameters for a redundant VT (rvt) interface the same way as you configure a VT (vt-) interface.

5. Add the redundant VT to the appropriate routing instance.

```
[edit routing-instances routing-instance-name]
user@host# set interface interface-name
```

For example:

```
[edit routing-instances pe-vrf]
user@host# set interface rvt0.0
```

NOTE: Because the rvt interface contains two or more vt- member interfaces, providing redundancy, adding an rvt interface and a vt- interface to the same routing instance is not supported.

See "Configuring Routing Instances on PE Routers in VPNs" on page 22 for detailed information about configuring routing instances.

SEE ALSO

Tunnel Interface Configuration on MX Series Routers Overview

Example: Configuring Redundant Virtual Tunnels to Provide Resiliency in Delivering Multicast Traffic

IN THIS SECTION

This example shows how to configure redundant virtual tunnels (VTs) in a multiprotocol BGP (MBGP) multicast VPN (MVPN). You configure a virtual loopback tunnel to facilitate virtual routing and forwarding (VRF) table lookup based on MPLS labels. Redundant VTs configured through this method provide near-immediate (less than 50 milliseconds) failover if one of the VTs fails.

Requirements

This example uses the following hardware and software components:

- MPCs on MX Series routers

- Junos OS Release 15.2

Overview

IN THIS SECTION

When a VT with an existing configuration joins a redundant VT, you must configure the redundant VT with the settings from the existing configuration.

You can add member VTs to a parent VT for redundancy.

On MX Series routers with MPCs, you can configure redundant VTs in these ways:

- You can create up to 16 redundant VTs.
- You can add up to 32 VTs as members of the redundant VTs.

> **NOTE**: The actual number of VTs you can create is determined by the type of chassis and the number of line cards you have.

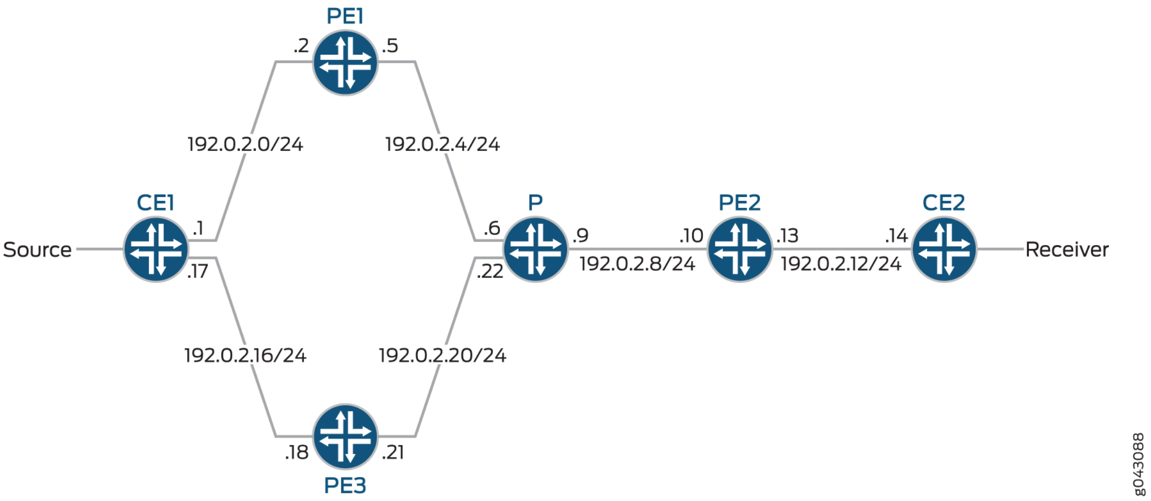
- When you add more than two VTs to a redundant VT, the members are in active mode by default.
- When you add only two members, you can configure the members in one of these ways:
  - Both members in active mode
  - One member in active mode and the other in backup mode

Topology

In this example, Device PE2 has a redundant VT interface configured in a multicast LDP routing instance. The redundant VT interface in this example contains two member VT interfaces, one in active mode and the other in backup mode.

Figure 61 on page 754 shows the topology used in this example.

Figure 61: Redundant VT Interfaces in MBGP MVPN



The following example shows the configuration for the customer edge (CE), provider (P), and provider edge (PE) devices in Figure 61 on page 754. See "Step-by-Step Procedure" on page 757 to configure Device PE2.

Configuration

IN THIS SECTION

Device P

```
set chassis network-services enhanced-ip
set interfaces ge-0/0/0 description “to PE1”
set interfaces ge-0/0/0 unit 0 family inet address 192.0.2.6/24
set interfaces ge-0/0/0 unit 0 family mpls
set interfaces ge-0/0/1 description “to PE2”
set interfaces ge-0/0/1 unit 0 family inet address 192.0.2.9/24
set interfaces ge-0/0/1 unit 0 family mpls
set interfaces ge-0/0/2 description “to PE3”
```

```
set interfaces ge-0/0/2 unit 0 family inet address 192.0.2.22/24
set interfaces ge-0/0/2 unit 0 family mpls
set interfaces lo0 unit 0 family inet address 198.51.100.3/24
set protocols rsvp interface all
set protocols rsvp interface fxp0.0 disable
set protocols mpls interface all
set protocols mpls interface fxp0.0 disable
set protocols bgp local-address 198.51.100.3
set protocols bgp family inet-vpn unicast
set protocols bgp group ibgp type internal
set protocols bgp group ibgp local-address 198.51.100.3
set protocols bgp group ibgp family inet any
set protocols bgp group ibgp family inet-vpn any
set protocols bgp group ibgp family inet-mvpn signaling
set protocols bgp group ibgp cluster 198.51.100.3
set protocols bgp group ibgp neighbor 198.51.100.2
set protocols bgp group ibgp neighbor 198.51.100.4
set protocols bgp group ibgp neighbor 198.51.100.6
set protocols ospf traffic-engineering
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set protocols ospf area 0.0.0.0 interface all
set protocols ospf area 0.0.0.0 interface fxp0.0 disable
set protocols ldp interface all
set protocols ldp interface fxp0.0 disable
set protocols pim interface all
set protocols pim interface fxp0.0 disable
set routing-options router-id 198.51.100.3
set routing-options autonomous-system 10
```

Device PE2

```
set chassis redundancy-group interface-type redundant-virtual-tunnel device-count 16
set chassis fpc 1 pic 0 tunnel-services bandwidth 1g
set chassis fpc 2 pic 0 tunnel-services bandwidth 1g
set chassis network-services enhanced-ip
set interfaces ge-0/0/0 description “to P”
set interfaces ge-0/0/0 unit 0 family inet address 192.0.2.10/24
set interfaces ge-0/0/0 unit 0 family mpls
set interfaces ge-0/0/1 description “to CE2”
set interfaces ge-0/0/1 unit 0 family inet address 192.0.2.13/24
set interfaces ge-0/0/1 unit 0 family mpls
set interfaces lo0 unit 0 family inet address 198.51.100.4/24
```

```
set interfaces rvt0 redundancy-group member-interface vt-1/0/10 active
set interfaces rvt0 redundancy-group member-interface vt-2/0/10 backup
set interfaces rvt0 unit 1000 family inet
set interfaces rvt0 unit 1000 family mpls
set protocols rsvp interface ge-0/0/0.0
set protocols rsvp interface fxp0.0 disable
set protocols mpls label-switched-path PE2-to-PE1 to 198.51.100.2
set protocols mpls label-switched-path PE2-to-PE3 to 198.51.100.6
set protocols mpls interface ge-0/0/0.0
set protocols mpls interface fxp0.0 disable
set protocols bgp group ibgp type internal
set protocols bgp group ibgp local-address 198.51.100.4
set protocols bgp group ibgp family inet any
set protocols bgp group ibgp family inet-vpn any
set protocols bgp group ibgp family inet-mvpn signaling
set protocols bgp group ibgp neighbor 198.51.100.3
set protocols ospf traffic-engineering
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set protocols ospf area 0.0.0.0 interface ge-0/0/0.0
set protocols ospf area 0.0.0.0 interface fxp0.0 disable
set protocols ldp interface all
set protocols ldp interface fxp0.0 disable
set protocols pim interface all
set protocols pim interface fxp0.0 disable
set policy-options policy-statement direct from protocol direct
set policy-options policy-statement direct then accept
set routing-instances vpn-1 instance-type vrf
set routing-instances vpn-1 interface ge-0/0/1.0
set routing-instances vpn-1 interface lo0.1
set routing-instances vpn-1 interface rvt0.1000
set routing-instances vpn-1 route-distinguisher 100:100
set routing-instances vpn-1 vrf-target target:1:1
set routing-instances vpn-1 routing-options multipath
set routing-instances vpn-1 protocols bgp group CE2-PE2 type external
set routing-instances vpn-1 protocols bgp group CE2-PE2 export direct
set routing-instances vpn-1 protocols bgp group CE2-PE2 peer-as 100
set routing-instances vpn-1 protocols bgp group CE2-PE2 neighbor 192.0.2.14 family inet unicast
set routing-instances vpn-1 protocols ospf area 0.0.0.0 interface all
set routing-instances vpn-1 protocols pim rp static address 192.168.0.0
set routing-instances vpn-1 protocols pim interface all mode sparse
set routing-instances vpn-1 protocols mvpn mvpn-mode spt-only
set routing-instances vpn-1 protocols mvpn sender-based-rpf
set routing-instances vpn-1 protocols mvpn hot-root-standby source-tree
```

```
set routing-options router-id 198.51.100.4
set routing-options autonomous-system 10
```

Device CE2

```
set  interfaces ge-0/0/0 description “upstream to PE2”
set  interfaces ge-0/0/0 unit 0 family inet address 192.0.2.14/24
set  interfaces ge-0/0/0 unit 0 family mpls
set  interfaces ge-0/0/1 description “Source-facing interface”
set  interfaces ge-0/0/1 unit 0 family inet address 203.0.113.2/24
set  interfaces lo0 unit 0 family inet address 198.51.100.5/24
set  protocols bgp group CE-to-PE type external
set  protocols bgp group CE-to-PE export direct
set  protocols bgp group CE-to-PE peer-as 10
set  protocols bgp group CE-to-PE local-as 100
set  protocols bgp group CE-to-PE neighbor 192.0.2.13 family inet unicast
set  protocols ospf area 0.0.0.0 interface lo0.0 passive
set  protocols ospf area 0.0.0.0 interface ge-0/0/0.0
set  protocols pim rp static address 192.168.0.0
set  protocols pim interface all
set  policy-options policy-statement direct from protocol direct
set  policy-options policy-statement direct then accept
set  routing-options router-id 198.51.100.5
set routing-options nonstop-routing
set routing-options autonomous-system 100
```

Procedure

Step-by-Step Procedure

The following example requires you to navigate various levels in the configuration hierarchy. For information about navigating the CLI, see *Using the CLI Editor in Configuration Mode* in the CLI User Guide.

In this section, the rvt interface is configured on Device PE2.

1. Configure the redundant virtual tunnel redundancy group and tunnel services on the chassis.

   ```
   [edit chassis]
   user@PE2# set redundancy-group interface-type redundant-virtual-tunnel device-count 16
   user@PE2# set fpc 1 pic 0 tunnel-services bandwidth 1g
   ```

```
user@PE2# set fpc 2 pic 0 tunnel-services bandwidth 1g
user@PE2# set network-services enhanced-ip
```

2. Configure the physical interfaces and loopback interfaces.

```
[edit interfaces]
user@PE2# set ge-0/0/0 description “to P”
user@PE2# set ge-0/0/0 unit 0 family inet address 192.0.2.10/24
user@PE2# set ge-0/0/0 unit 0 family mpls
user@PE2# set ge-0/0/1 description “to CE2”
user@PE2# set ge-0/0/1 unit 0 family inet address 192.0.2.13/24
user@PE2# set ge-0/0/1 unit 0 family mpls
user@PE2# set lo0 unit 0 family inet address 198.51.100.4/24
```

3. Configure the rvt interface.

NOTE: In this example, one member of the rvt interface is set to be active and the other interface is set to be the backup. This approach requires member interfaces to be on separate MPCs.

```
[edit interfaces]
user@PE2# set rvt0 redundancy-group member-interface vt-1/0/10 active
user@PE2# set rvt0 redundancy-group member-interface vt-2/0/10 backup
user@PE2# set rvt0 unit 1000 family inet
user@PE2# set rvt0 unit 1000 family mpls
```

4. Configure MPLS.

```
[edit protocols mpls]
user@PE2# set label-switched-path PE2-to-PE1 to 198.51.100.2
user@PE2# set label-switched-path PE2-to-PE3 to 198.51.100.6
user@PE2# set interface ge-0/0/0.0
user@PE2# set interface fxp0.0 disable
```

5. Configure BGP.

```
[edit protocols bgp]
user@PE2# set group ibgp type internal
user@PE2# set group ibgp local-address 198.51.100.4
user@PE2# set group ibgp family inet any
user@PE2# set group ibgp family inet-vpn any
user@PE2# set group ibgp family inet-mvpn signaling
user@PE2# set group ibgp neighbor 198.51.100.3
```

6. Configure an interior gateway protocol.

```
[edit  protocols ospf]
user@PE2# set traffic-engineering
user@PE2# set area 0.0.0.0 interface lo0.0 passive
user@PE2# set area 0.0.0.0 interface ge-0/0/0.0
user@PE2# set area 0.0.0.0 interface fxp0.0 disable
```

7. Configure LDP, PIM, and RSVP.

```
[edit protocols]
user@PE2# set ldp interface all
user@PE2# set ldp interface fxp0.0 disable
user@PE2# set pim interface all
user@PE2# set pim interface fxp0.0 disable
user@PE2# set rsvp interface ge-0/0/0.0
user@PE2# set rsvp interface fxp0.0 disable
```

8. Configure the routing policy.

```
[edit policy-options policy-statement direct]
user@PE2# set from protocol direct
user@PE2# set then accept
```

9. Configure the routing instance.

```
[edit routing-instances vpn-1]
user@PE2# set instance-type vrf
```

```
user@PE2# set interface ge-0/0/1.0
user@PE2# set interface lo0.1
user@PE2# set route-distinguisher 100:100
user@PE2# set vrf-target target:1:1
user@PE2# set routing-options multipath
user@PE2# set protocols bgp group CE2-PE2 type external
user@PE2# set protocols bgp group CE2-PE2 export direct
user@PE2# set protocols bgp group CE2-PE2 peer-as 100
user@PE2# set protocols bgp group CE2-PE2 neighbor 192.0.2.14 family inet unicast
user@PE2# set protocols ospf area 0.0.0.0 interface all
user@PE2# set protocols pim rp static address 192.168.0.0
user@PE2# set protocols pim interface all mode sparse
user@PE2# set protocols mvpn mvpn-mode spt-only
user@PE2# set protocols mvpn sender-based-rpf
user@PE2# set protocols mvpn hot-root-standby source-tree
```

10. Add the rvt interface to the routing instance.

```
[edit routing-instances vpn-1]
user@PE2# set interface rvt0.1000
```

11. Configure the router ID and autonomous system (AS) number.

```
[edit  routing-options]
user@PE2# set router-id 198.51.100.4
user@PE2# set autonomous-system 10
```

Results

From configuration mode, confirm your configuration by entering the **show chassis**, **show interfaces**, **show protocols**, **show policy-options**, **show routing-instances**, and **show routing-options** commands. If the output does not display the intended configuration, repeat the configuration instructions in this example to correct it.

```
user@PE2# show chassis
redundancy-group {
    interface-type {
        redundant-virtual-tunnel {
            device-count 16;
```

```
        }
    }
}
fpc 1 {
    pic 0 {
        tunnel-services {
            bandwidth 1g;
        }
    }
}
fpc 2 {
    pic 0 {
        tunnel-services {
            bandwidth 1g;
        }
    }
}
network-services {
    enhanced-ip;
}
user@PE2# show interfaces
ge-0/0/0 {
    description "to P";
    unit 0 {
        family inet {
            address 192.0.2.10/24;
        }
        family mpls;
    }
}
ge-0/0/1 {
    description "to CE2";
    unit 0 {
        family inet {
            address 192.0.2.13/24;
        }
        family mpls;
    }
}
lo0 {
    unit 0 {
        family inet {
            address 198.51.100.4/24;
```

```
        }
    }
}
rvt0 {
    redundancy-group {
        member-interface vt-1/0/10 {
            active;
        }
        member-interface vt-2/0/10 {
            backup;
        }
    }
    unit 1000 {
        family inet;
        family mpls;
    }
}
user@PE2# show protocols
mpls {
    label-switched-path PE2-to-PE1 {
        to 198.51.100.2;
    }
    label-switched-path PE2-to-PE3 {
        to 198.51.100.6;
    }
    interface fxp0.0 {
        disable;
    }
    interface ge-0/0/0.0;
}
bgp {
    group ibgp {
        type internal;
        local-address 198.51.100.4;
        family inet {
            any;
        }
        family inet-vpn {
            any;
        }
        family inet-mvpn {
            signaling;
        }
```

```
        neighbor 198.51.100.3;
    }
}
ospf {
    traffic-engineering;
    area 0.0.0.0 {
        interface fxp0.0 {
            disable;
        }
        interface lo0.0 {
            passive;
        }
        interface ge-0/0/0.0;
    }
}
ldp {
    interface all;
    interface fxp0.0 {
        disable;
    }
}
pim {
    interface all;
    interface fxp0.0 {
        disable;
    }
}
rsvp {
    interface fxp0.0 {
        disable;
    }
    interface ge-0/0/0.0;
}

user@PE2# show policy-options
policy-statement direct {
    from protocol direct;
    then accept;
}
user@PE2# show routing-instances
vpn-1 {
    instance-type vrf;
    interface ge-0/0/1.0;
```

```
    interface lo0.1;
    interface rvt0.1000;
    route-distinguisher 100:100;
    vrf-target target:1:1;
    routing-options {
        multipath;
    }
    protocols {
        bgp {
            group CE2-PE2 {
                type external;
                export direct;
                peer-as 100;
                neighbor 192.0.2.14 {
                    family inet {
                        unicast;
                    }
                }
            }
        }
        ospf {
            area 0.0.0.0 {
                interface all;
            }
        }
        pim {
            rp {
                static {
                    address 192.168.0.0;
                }
            }
            interface all {
                mode sparse;
            }
        }
        mvpn {
            mvpn-mode {
                spt-only;
            }
            sender-based-rpf;
            hot-root-standby {
                source-tree;
            }
```

```
        }
    }
}
user@PE2# show routing-options
router-id 198.51.100.4;
autonomous-system 10;
```

If you are done configuring the device, enter **commit** from configuration mode.

Verification

IN THIS SECTION

Confirm that the configuration is working properly.

Verifying the Creation of Virtual Tunnel and Redundant Virtual Tunnel Interfaces

Purpose

Verify that the rvt interface is created and that the appropriate member vt- interfaces are contained within the rvt interface.

Action

From operational mode, enter the **show interfaces terse | match rvt0** command.

```
user@PE2# show interfaces terse | match rvt0
user@host# run show interfaces terse | match rvt0
vt-1/0/10.1000          up    up   container--> rvt0.1000
vt-2/0/10.1000          up    up   container--> rvt0.1000
rvt0                    up    up
rvt0.1000               up    up   inet
```

Meaning

The output shows that **rvt0** was created and that it contains **vt-1/0/10** and **vt-2/0/10** as member interfaces.

Verifying the Inclusion of the rvt Interface in the Multicast Route

Purpose

Verify that the **vpn1** multicast routing instance is running and that it contains the **rvt0** interface.

Action

From operational mode, enter the **show multicast route extensive instance vpn1** command.

```
 user@PE2# show multicast route extensive instance vpn1
Instance: vpn1 Family: INET

Group: 203.0.113.3
    Source: 10.0.0.2/24
    Upstream rpf interface list:
        rvt0.1000 (P)
            Sender Id: Label 299888
    Downstream interface list:
        ge-1/0/5.0
    Number of outgoing interfaces: 1
    Session description: Unknown
    Statistics: 460 kBps, 10000 pps, 4387087 packets
    RPF Next-hop ID: 941
    Next-hop ID: 1048594
    Upstream protocol: MVPN
    Route state: Active
    Forwarding state: Forwarding
    Cache lifetime/timeout: forever
    Wrong incoming interface notifications: 0
    Uptime: 00:07:48
```

Meaning

The output shows that the **vpn1** routing instance is running and contains **rvt0**.

Verifying Traffic Through the rvt Interface and Its Member Interfaces

Purpose

Verify that traffic passes through the **rvt0** interface and its member interfaces as expected. For this section, start streams of unicast and multicast traffic from the source to the receiver.

> **NOTE**: In this example, all traffic through the **rvt0** interface is expected to pass through the active **vt-1/0/10** interface, and no traffic is expected to pass through the backup **vt-2/0/10** interface.

Action

From operational mode, enter the **monitor interface rvt0** command.

```
 user@PE2# monitor interface rvt0
Interface: rvt0, Enabled, Link is Up
Encapsulation: VPN-Loopback-tunnel, Speed: 2000mbps
Traffic statistics:                                              Current delta
  Input bytes:                2788575982 (14720096 bps)             [10976152]
  Output bytes:                        0 (0 bps)                           [0]
  Input packets:                60621217 (40000 pps)                  [238612]
  Output packets:                      0 (0 pps)                           [0]
Error statistics:
  Input errors:                        0                                   [0]
  Input drops:                         0                                   [0]
  Input framing errors:                0                                   [0]
  Carrier transitions:                 0                                   [0]
  Output errors:                       0                                   [0]
  Output drops:                        0                                   [0]
```

From this output, **rvt0** is enabled and up and traffic is flowing through it.

Now enter the **monitor interface vt-1/0/10** command.

```
user@PE2# monitor interface vt-1/0/10
Interface: vt-1/0/10, Enabled, Link is Up
Encapsulation: VPN-Loopback-tunnel, Speed: 1000mbps
Traffic statistics:                                              Current delta
  Input bytes:                2997120202 (14720096 bps)              [3658702]
```

```
  Output bytes:                  0 (0 bps)                           [0]
  Input packets:          65154787 (40000 pps)                   [79537]
  Output packets:                0 (0 pps)                           [0]
Error statistics:
  Input errors:                  0                                   [0]
  Input drops:                   0                                   [0]
  Input framing errors:          0                                   [0]
  Carrier transitions:           0                                   [0]
  Output errors:                 0                                   [0]
  Output drops:                  0                                   [0]
```

From this output, all traffic that is passing through the **rvt0** interface is passing through the active **vt-1/0/10** interface, as expected.

Now enter the **monitor interface vt-2/0/10** command.

```
 user@PE2# monitor interface vt-2/0/10
Interface: vt-2/0/10, Enabled, Link is Up
Encapsulation: VPN-Loopback-tunnel, Speed: 1000mbps
Traffic statistics:                                         Current delta
  Input bytes:                   0 (0 bps)                           [0]
  Output bytes:                  0 (0 bps)                           [0]
  Input packets:                 0 (0 pps)                           [0]
  Output packets:                0 (0 pps)                           [0]
Error statistics:
  Input errors:                  0                                   [0]
  Input drops:                   0                                   [0]
  Input framing errors:          0                                   [0]
  Carrier transitions:           0                                   [0]
  Output errors:                 0                                   [0]
  Output drops:                  0                                   [0]
```

From this output, even though **vt-2/0/10** is enabled and up, no traffic is passing through. This is expected, because the interface is set to **backup**.

Meaning

While everything is running normally, traffic does pass through the redundant virtual interface, **rvt0**, and all of the traffic passes through its active member interface. If neither member interface is set to be **active** or **backup**, the traffic is expected to be load balanced across both interfaces.

Verifying Immediate Failover to the Backup Virtual Tunnel

Purpose

Verify that when the MPC containing the active member interface fails or is restarted, all traffic through the **rvt0** interfaces immediately fails over to the backup member interface.

> **BEST PRACTICE**: For this task, we recommend you have one window open showing the live statistics of the backup interface, through the **monitor interface vt-2/0/10** command, and another window from which you can restart the MPC containing the active member interface.

Action

From operational mode, enter the **request chassis fpc slot 1 restart** command and observe the live statistics of the backup member interface.

```
 user@PE2# request chassis fpc slot 1 restart
Interface: vt-2/0/10, Enabled, Link is Up
Encapsulation: VPN-Loopback-tunnel, Speed: 1000mbps
Traffic statistics:                                              Current delta
  Input bytes:                 354964428 (14720248 bps)             [13220676]
  Output bytes:                        0 (0 bps)                           [0]
  Input packets:                 7716618 (40000 pps)                  [287406]
  Output packets:                      0 (0 pps)                           [0]
Error statistics:
  Input errors:                        0                                   [0]
  Input drops:                         0                                   [0]
  Input framing errors:                0                                   [0]
  Carrier transitions:                 0                                   [0]
  Output errors:                       0                                   [0]
  Output drops:                        0                                   [0]
```

Meaning

The output shows that the traffic through the **rvt0** interface is immediately fully carried by the backup member interface.

Understanding Redundant Virtual Tunnel Interfaces in MBGP MVPNs

In multiprotocol BGP (MBGP) multicast VPNs (MVPNs), VT interfaces are needed for multicast traffic on routing devices that function as combined provider edge (PE) and provider core (P) routers to optimize bandwidth usage on core links. VT interfaces prevent traffic replication when a P router also acts as a PE router (an exit point for multicast traffic).

Starting in Junos OS Release 12.3, you can configure up to eight VT interfaces in a routing instance, thus providing Tunnel PIC redundancy inside the same multicast VPN routing instance. When the active VT interface fails, the secondary one takes over, and you can continue managing multicast traffic with no duplication.

Redundant VT interfaces are supported with RSVP point-to-multipoint provider tunnels as well as multicast LDP provider tunnels. This feature also works for extranets.

You can configure one of the VT interfaces to be the primary interface. If a VT interface is configured as the primary, it becomes the next hop that is used for traffic coming in from the core on the label-switched path (LSP) into the routing instance. When a VT interface is configured to be primary and the VT interface is used for both unicast and multicast traffic, only the multicast traffic is affected.

If no VT interface is configured to be the primary or if the primary VT interface is unusable, one of the usable configured VT interfaces is chosen to be the next hop that is used for traffic coming in from the core on the LSP into the routing instance. If the VT interface in use goes down for any reason, another usable configured VT interface in the routing instance is chosen. When the VT interface in use changes, all multicast routes in the instance also switch their reverse-path forwarding (RPF) interface to the new VT interface to allow the traffic to be received.

To realize the full benefit of redundancy, we recommend that when you configure multiple VT interfaces, at least one of the VT interfaces be on a different Tunnel PIC from the other VT interfaces. However, Junos OS does not enforce this.

Example: Configuring Redundant Virtual Tunnel Interfaces in MBGP MVPNs

IN THIS SECTION

This example shows how to configure redundant virtual tunnel (VT) interfaces in multiprotocol BGP (MBGP) multicast VPNs (MVPNs). To configure, include multiple VT interfaces in the routing instance and, optionally, apply the `primary` statement to one of the VT interfaces.

Requirements

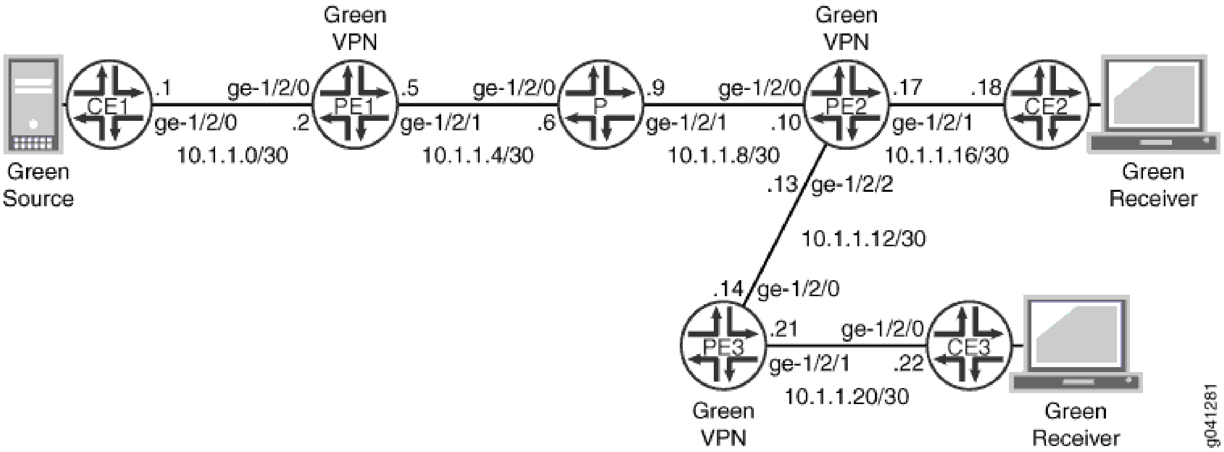
The routing device that has redundant VT interfaces configured must be running Junos OS Release 12.3 or later.

Overview

In this example, Device PE2 has redundant VT interfaces configured in a multicast LDP routing instance, and one of the VT interfaces is assigned to be the primary interface.

Figure 62 on page 771 shows the topology used in this example.

Figure 62: Multiple VT Interfaces in MBGP MVPN Topology



The following example shows the configuration for the customer edge (CE), provider (P), and provider edge (PE) devices in Figure 62 on page 771. The section "Step-by-Step Procedure" on page 776 describes the steps on Device PE2.

Configuration

IN THIS SECTION

Procedure

CLI Quick Configuration

To quickly configure this example, copy the following commands, paste them into a text file, remove any line breaks, change any details necessary to match your network configuration, and then copy and paste the commands into the CLI at the `[edit]` hierarchy level.

Device CE1

```
set interfaces ge-1/2/0 unit 0 family inet address 10.1.1.1/30
set interfaces ge-1/2/0 unit 0 family mpls
set interfaces lo0 unit 0 family inet address 192.0.2.1/24
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set protocols ospf area 0.0.0.0 interface ge-1/2/0.0
set protocols pim rp static address 198.51.100.0
set protocols pim interface all
set routing-options router-id 192.0.2.1
```

Device CE2

```
set  interfaces ge-1/2/0 unit 0 family inet address 10.1.1.18/30
set  interfaces ge-1/2/0 unit 0 family mpls
set  interfaces lo0 unit 0 family inet address 192.0.2.6/24
set  protocols sap listen 192.168.0.0
set  protocols ospf area 0.0.0.0 interface lo0.0 passive
set  protocols ospf area 0.0.0.0 interface ge-1/2/0.0
set  protocols pim rp static address 198.51.100.0
set  protocols pim interface all
set  routing-options router-id 192.0.2.6
```

Device CE3

```
set interfaces ge-1/2/0 unit 0 family inet address 10.1.1.22/30
set interfaces ge-1/2/0 unit 0 family mpls
set interfaces lo0 unit 0 family inet address 192.0.2.7/24
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set protocols ospf area 0.0.0.0 interface ge-1/2/0.0
set protocols pim rp static address 198.51.100.0
```

```
set protocols pim interface all
set routing-options router-id 192.0.2.7
```

Device P

```
set interfaces ge-1/2/0 unit 0 family inet address 10.1.1.6/30
set interfaces ge-1/2/0 unit 0 family mpls
set interfaces ge-1/2/1 unit 0 family inet address 10.1.1.9/30
set interfaces ge-1/2/1 unit 0 family mpls
set interfaces lo0 unit 0 family inet address 192.0.2.3/24
set protocols mpls interface ge-1/2/0.0
set protocols mpls interface ge-1/2/1.0
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set protocols ospf area 0.0.0.0 interface ge-1/2/0.0
set protocols ospf area 0.0.0.0 interface ge-1/2/1.0
set protocols ldp interface ge-1/2/0.0
set protocols ldp interface ge-1/2/1.0
set protocols ldp p2mp
set routing-options router-id 192.0.2.3
```

Device PE1

```
set interfaces ge-1/2/0 unit 0 family inet address 10.1.1.2/30
set interfaces ge-1/2/0 unit 0 family mpls
set interfaces ge-1/2/1 unit 0 family inet address 10.1.1.5/30
set interfaces ge-1/2/1 unit 0 family mpls
set interfaces vt-1/2/0 unit 2 family inet
set interfaces lo0 unit 0 family inet address 192.0.2.2/24
set interfaces lo0 unit 1 family inet address 198.51.100.0/24
set protocols mpls interface ge-1/2/1.0
set protocols bgp group ibgp type internal
set protocols bgp group ibgp local-address 192.0.2.2
set protocols bgp group ibgp family inet-vpn any
set protocols bgp group ibgp family inet-mvpn signaling
set protocols bgp group ibgp neighbor 192.0.2.4
set protocols bgp group ibgp neighbor 192.0.2.5
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set protocols ospf area 0.0.0.0 interface ge-1/2/1.0
set protocols ldp interface ge-1/2/1.0
set protocols ldp p2mp
set policy-options policy-statement parent_vpn_routes from protocol bgp
```

```
set policy-options policy-statement parent_vpn_routes then accept
set routing-instances vpn-1 instance-type vrf
set routing-instances vpn-1 interface ge-1/2/0.0
set routing-instances vpn-1 interface vt-1/2/0.2 multicast
set routing-instances vpn-1 interface lo0.1
set routing-instances vpn-1 route-distinguisher 100:100
set routing-instances vpn-1 provider-tunnel ldp-p2mp
set routing-instances vpn-1 vrf-target target:1:1
set routing-instances vpn-1 protocols ospf export parent_vpn_routes
set routing-instances vpn-1 protocols ospf area 0.0.0.0 interface lo0.1 passive
set routing-instances vpn-1 protocols ospf area 0.0.0.0 interface ge-1/2/0.0
set routing-instances vpn-1 protocols pim rp static address 198.51.100.0
set routing-instances vpn-1 protocols pim interface ge-1/2/0.0 mode sparse
set routing-instances vpn-1 protocols mvpn
set routing-options router-id 192.0.2.2
set routing-options autonomous-system 1001
```

Device PE2

```
set interfaces ge-1/2/0 unit 0 family inet address 10.1.1.10/30
set interfaces ge-1/2/0 unit 0 family mpls
set interfaces ge-1/2/2 unit 0 family inet address 10.1.1.13/30
set interfaces ge-1/2/2 unit 0 family mpls
set interfaces ge-1/2/1 unit 0 family inet address 10.1.1.17/30
set interfaces ge-1/2/1 unit 0 family mpls
set interfaces vt-1/1/0 unit 0 family inet
set interfaces vt-1/2/1 unit 0 family inet
set interfaces lo0 unit 0 family inet address 192.0.2.4/24
set interfaces lo0 unit 1 family inet address 203.0.113.4/24
set protocols mpls interface ge-1/2/0.0
set protocols mpls interface ge-1/2/2.0
set protocols bgp group ibgp type internal
set protocols bgp group ibgp local-address 192.0.2.4
set protocols bgp group ibgp family inet-vpn any
set protocols bgp group ibgp family inet-mvpn signaling
set protocols bgp group ibgp neighbor 192.0.2.2
set protocols bgp group ibgp neighbor 192.0.2.5
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set protocols ospf area 0.0.0.0 interface ge-1/2/0.0
set protocols ospf area 0.0.0.0 interface ge-1/2/2.0
set protocols ldp interface ge-1/2/0.0
set protocols ldp interface ge-1/2/2.0
```

```
set protocols ldp p2mp
set policy-options policy-statement parent_vpn_routes from protocol bgp
set policy-options policy-statement parent_vpn_routes then accept
set routing-instances vpn-1 instance-type vrf
set routing-instances vpn-1 interface vt-1/1/0.0 multicast
set routing-instances vpn-1 interface vt-1/1/0.0 primary
set routing-instances vpn-1 interface vt-1/2/1.0 multicast
set routing-instances vpn-1 interface ge-1/2/1.0
set routing-instances vpn-1 interface lo0.1
set routing-instances vpn-1 route-distinguisher 100:100
set routing-instances vpn-1 vrf-target target:1:1
set routing-instances vpn-1 protocols ospf export parent_vpn_routes
set routing-instances vpn-1 protocols ospf area 0.0.0.0 interface lo0.1 passive
set routing-instances vpn-1 protocols ospf area 0.0.0.0 interface ge-1/2/1.0
set routing-instances vpn-1 protocols pim rp static address 198.51.100.0
set routing-instances vpn-1 protocols pim interface ge-1/2/1.0 mode sparse
set routing-instances vpn-1 protocols mvpn
set routing-options router-id 192.0.2.4
set routing-options autonomous-system 1001
```

Device PE3

```
set interfaces ge-1/2/0 unit 0 family inet address 10.1.1.14/30
set interfaces ge-1/2/0 unit 0 family mpls
set interfaces ge-1/2/1 unit 0 family inet address 10.1.1.21/30
set interfaces ge-1/2/1 unit 0 family mpls
set interfaces vt-1/2/0 unit 5 family inet
set interfaces lo0 unit 0 family inet address 192.0.2.5/24
set interfaces lo0 unit 1 family inet address 203.0.113.5/24
set protocols mpls interface ge-1/2/0.0
set protocols bgp group ibgp type internal
set protocols bgp group ibgp local-address 192.0.2.5
set protocols bgp group ibgp family inet-vpn any
set protocols bgp group ibgp family inet-mvpn signaling
set protocols bgp group ibgp neighbor 192.0.2.2
set protocols bgp group ibgp neighbor 192.0.2.4
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set protocols ospf area 0.0.0.0 interface ge-1/2/0.0
set protocols ldp interface ge-1/2/0.0
set protocols ldp p2mp
set policy-options policy-statement parent_vpn_routes from protocol bgp
set policy-options policy-statement parent_vpn_routes then accept
```

```
set routing-instances vpn-1 instance-type vrf
set routing-instances vpn-1 interface vt-1/2/0.5 multicast
set routing-instances vpn-1 interface ge-1/2/1.0
set routing-instances vpn-1 interface lo0.1
set routing-instances vpn-1 route-distinguisher 100:100
set routing-instances vpn-1 vrf-target target:1:1
set routing-instances vpn-1 protocols ospf export parent_vpn_routes
set routing-instances vpn-1 protocols ospf area 0.0.0.0 interface lo0.1 passive
set routing-instances vpn-1 protocols ospf area 0.0.0.0 interface ge-1/2/1.0
set routing-instances vpn-1 protocols pim rp static address 198.51.100.0
set routing-instances vpn-1 protocols pim interface ge-1/2/1.0 mode sparse
set routing-instances vpn-1 protocols mvpn
set routing-options router-id 192.0.2.5
set routing-options autonomous-system 1001
```

Step-by-Step Procedure

The following example requires you to navigate various levels in the configuration hierarchy. For information about navigating the CLI, see *Using the CLI Editor in Configuration Mode* in the CLI User Guide.

To configure redundant VT interfaces in an MBGP MVPN:

1. Configure the physical interfaces and loopback interfaces.

```
[edit interfaces]
user@PE2# set ge-1/2/0 unit 0 family inet address 10.1.1.10/30
user@PE2# set ge-1/2/0 unit 0 family mpls
user@PE2# set ge-1/2/2 unit 0 family inet address 10.1.1.13/30
user@PE2# set ge-1/2/2 unit 0 family mpls
user@PE2# set ge-1/2/1 unit 0 family inet address 10.1.1.17/30
user@PE2# set ge-1/2/1 unit 0 family mpls
user@PE2# set lo0 unit 0 family inet address 192.0.2.4/24
user@PE2# set lo0 unit 1 family inet address 203.0.113.4/24
```

2. Configure the VT interfaces.

Each VT interface is configurable under one routing instance.

```
[edit interfaces]
user@PE2# set vt-1/1/0 unit 0 family inet
user@PE2# set vt-1/2/1 unit 0 family inet
```

3. Configure MPLS on the physical interfaces.

```
[edit  protocols mpls]
user@PE2# set interface ge-1/2/0.0
user@PE2# set interface ge-1/2/2.0
```

4. Configure BGP.

```
[edit  protocols bgp group ibgp]
user@PE2# set type internal
user@PE2# set local-address 192.0.2.4
user@PE2# set family inet-vpn any
user@PE2# set family inet-mvpn signaling
user@PE2# set neighbor 192.0.2.2
user@PE2# set neighbor 192.0.2.5
```

5. Configure an interior gateway protocol.

```
[edit  protocols ospf area 0.0.0.0]
user@PE2# set interface lo0.0 passive
user@PE2# set interface ge-1/2/0.0
user@PE2# set interface ge-1/2/2.0
```

6. Configure LDP.

```
[edit  protocols ldp]
user@PE2# set interface ge-1/2/0.0
user@PE2# set interface ge-1/2/2.0
user@PE2# set p2mp
```

7. Configure the routing policy.

```
[edit  policy-options policy-statement parent_vpn_routes]
user@PE2# set from protocol bgp
user@PE2# set then accept
```

8. Configure the routing instance.

```
[edit  routing-instances vpn-1]
user@PE2# set instance-type vrf
user@PE2# set interface ge-1/2/1.0
user@PE2# set interface lo0.1
user@PE2# set route-distinguisher 100:100
user@PE2# set vrf-target target:1:1
user@PE2# set protocols ospf export parent_vpn_routes
user@PE2# set protocols ospf area 0.0.0.0 interface lo0.1 passive
user@PE2# set protocols ospf area 0.0.0.0 interface ge-1/2/1.0
user@PE2# set protocols pim rp static address 198.51.100.0
user@PE2# set protocols pim interface ge-1/2/1.0 mode sparse
user@PE2# set protocols mvpn
```

9. Configure redundant VT interfaces in the routing instance.

   Make vt-1/1/0.0 the primary interface.

```
[edit  routing-instances vpn-1]
user@PE2# set interface vt-1/1/0.0 multicast primary
user@PE2# set interface vt-1/2/1.0 multicast
```

10. Configure the router ID and autonomous system (AS) number.

```
[edit  routing-options]
user@PE2# set router-id 192.0.2.4
user@PE2# set autonomous-system 1001
```

Results

From configuration mode, confirm your configuration by entering the **show interfaces**, **show protocols**, **show policy-options**, **show routing-instances**, and **show routing-options** commands. If the output does not display the intended configuration, repeat the configuration instructions in this example to correct it.

```
user@PE2# show interfaces
ge-1/2/0 {
    unit 0 {
        family inet {
            address 10.1.1.10/30;
        }
        family mpls;
    }
}
ge-1/2/2 {
    unit 0 {
        family inet {
            address 10.1.1.13/30;
        }
        family mpls;
    }
}
ge-1/2/1 {
    unit 0 {
        family inet {
            address 10.1.1.17/30;
        }
        family mpls;
    }
}
vt-1/1/0 {
    unit 0 {
        family inet;
    }
}
vt-1/2/1 {
    unit 0 {
        family inet;
    }
}
lo0 {
```

```
    unit 0 {
        family inet {
            address 192.0.2.4/24;
        }
    }
    unit 1 {
        family inet {
            address 203.0.113.4/24;
        }
    }
}
```

```
user@PE2# show protocols
mpls {
    interface ge-1/2/0.0;
    interface ge-1/2/2.0;
}
bgp {
    group ibgp {
        type internal;
        local-address 192.0.2.4;
        family inet-vpn {
            any;
        }
        family inet-mvpn {
            signaling;
        }
        neighbor 192.0.2.2;
        neighbor 192.0.2.5;
    }
}
ospf {
    area 0.0.0.0 {
        interface lo0.0 {
            passive;
        }
        interface ge-1/2/0.0;
        interface ge-1/2/2.0;
    }
}
ldp {
```

```
    interface ge-1/2/0.0;
    interface ge-1/2/2.0;
    p2mp;
}
```

```
user@PE2# show policy-options
policy-statement parent_vpn_routes {
    from protocol bgp;
    then accept;
}
```

```
user@PE2# show routing-instances
vpn-1 {
    instance-type vrf;
    interface vt-1/1/0.0 {
        multicast;
        primary;
    }
    interface vt-1/2/1.0 {
        multicast;
    }
    interface ge-1/2/1.0;
    interface lo0.1;
    route-distinguisher 100:100;
    vrf-target target:1:1;
    protocols {
        ospf {
            export parent_vpn_routes;
            area 0.0.0.0 {
                interface lo0.1 {
                    passive;
                }
                interface ge-1/2/1.0;
            }
        }
        pim {
            rp {
                static {
                    address 198.51.100.0;
                }
```

```
            }
            interface ge-1/2/1.0 {
                mode sparse;
            }
        }
        mvpn;
    }
}
```

```
user@PE2# show routing-options
router-id 192.0.2.4;
autonomous-system 1001;
```

If you are done configuring the device, enter **commit** from configuration mode.

Verification

IN THIS SECTION

Confirm that the configuration is working properly.

NOTE: The `show multicast route extensive instance` *`instance-name`* command also displays the VT interface in the multicast forwarding table when multicast traffic is transmitted across the VPN.

Checking the LSP Route

Purpose

Verify that the expected LT interface is assigned to the LDP-learned route.

Action

1. From operational mode, enter the **show route table mpls** command.

```
user@PE2> show route table mpls
mpls.0: 13 destinations, 13 routes (13 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

0                  *[MPLS/0] 02:09:36, metric 1
                         Receive
1                  *[MPLS/0] 02:09:36, metric 1
                         Receive
2                  *[MPLS/0] 02:09:36, metric 1
                         Receive
13                 *[MPLS/0] 02:09:36, metric 1
                         Receive
299776             *[LDP/9] 02:09:14, metric 1
                    >    via ge-1/2/0.0, Pop
299776(S=0)        *[LDP/9] 02:09:14, metric 1
                    >    via ge-1/2/0.0, Pop
299792             *[LDP/9] 02:09:09, metric 1
                    >    via ge-1/2/2.0, Pop
299792(S=0)        *[LDP/9] 02:09:09, metric 1
                    >    via ge-1/2/2.0, Pop
299808             *[LDP/9] 02:09:04, metric 1
                    >    via ge-1/2/0.0, Swap 299808
299824             *[VPN/170] 02:08:56
                    >    via ge-1/2/1.0, Pop
299840             *[VPN/170] 02:08:56
                    >    via ge-1/2/1.0, Pop
299856             *[VPN/170] 02:08:56
                      receive table vpn-1.inet.0, Pop
299872             *[LDP/9] 02:08:54, metric 1
                    >    via vt-1/1/0.0, Pop
                         via ge-1/2/2.0, Swap 299872
```

2. From configuration mode, change the primary VT interface by removing the `primary` statement from the vt-1/1/0.0 interface and adding it to the vt-1/2/1.0 interface.

```
[edit routing-instances vpn-1]
user@PE2#  delete interface vt-1/1/0.0 primary
```

```
user@PE2#  set interface vt-1/2/1.0 primary
user@PE2# commit
```

3. From operational mode, enter the **show route table mpls** command.

```
user@PE2> show route table mpls
mpls.0: 13 destinations, 13 routes (13 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

0                  *[MPLS/0] 02:09:36, metric 1
                         Receive
1                  *[MPLS/0] 02:09:36, metric 1
                         Receive
2                  *[MPLS/0] 02:09:36, metric 1
                         Receive
13                 *[MPLS/0] 02:09:36, metric 1
                         Receive
299776             *[LDP/9] 02:09:14, metric 1
                    >    via ge-1/2/0.0, Pop
299776(S=0)        *[LDP/9] 02:09:14, metric 1
                    >    via ge-1/2/0.0, Pop
299792             *[LDP/9] 02:09:09, metric 1
                    >    via ge-1/2/2.0, Pop
299792(S=0)        *[LDP/9] 02:09:09, metric 1
                    >    via ge-1/2/2.0, Pop
299808             *[LDP/9] 02:09:04, metric 1
                    >    via ge-1/2/0.0, Swap 299808
299824             *[VPN/170] 02:08:56
                    >    via ge-1/2/1.0, Pop
299840             *[VPN/170] 02:08:56
                    >    via ge-1/2/1.0, Pop
299856             *[VPN/170] 02:08:56
                      receive table vpn-1.inet.0, Pop
299872             *[LDP/9] 02:08:54, metric 1
                    >    via vt-1/2/1.0, Pop
                         via ge-1/2/2.0, Swap 299872
```

Meaning

With the original configuration, the output shows the vt-1/1/0.0 interface. If you change the primary interface to vt-1/2/1.0, the output shows the vt-1/2/1.0 interface.

Change History Table

Feature support is determined by the platform and release you are using. Use Feature Explorer to determine if a feature is supported on your platform.

Release	Description
12.3	Starting in Junos OS Release 12.3, you can configure up to eight VT interfaces in a routing instance, thus providing Tunnel PIC redundancy inside the same multicast VPN routing instance.

MVPN VRF Import and Export Policies

IN THIS SECTION

Configuring VRF Route Targets for Routing Instances for an MBGP MVPN

IN THIS SECTION

By default, the VPN routing and forwarding (VRF) import and export route targets (configured either using VRF import and export policies or using the `vrf-target` statement) are used for importing and exporting routes with the MBGP MVPN network layer reachability information (NLRI).

You can use the `export-target` and `import-target` statements to override the default VRF import and export route targets. Export and import targets can also be specified specifically for sender sites or receiver sites, or can be borrowed from a configured unicast route target. Note that a sender site export route target is always advertised when security association routes are exported.

NOTE: When you configure an MBGP MVPN routing instance, you should not configure a target value for an MBGP MVPN specific route target that is identical to a target value for a unicast route target configured in another routing instance.

Specifying route targets in the MBGP MVPN NLRI for sender and receiver sites is useful when there is a mix of sender only, receiver only, and sender and receiver sites. A sender site route target is used for exporting automatic discovery routes by a sender site and for importing automatic discovery routes by a receiver site. A receiver site route target is used for exporting routes by a receiver site and importing routes by a sender site. A sender and receiver site exports and imports routes with both route targets.

A provider edge (PE) router with sites in a specific MBGP MVPN must determine whether a received automatic discovery route is from a sender site or receiver site based on the following:

- If the PE router is configured to be only in a sender site, route targets are imported only from receiver sites. Imported automatic discovery routes must be from a receiver site.
- If the PE router is configured to be only in a receiver site, route targets are imported only from sender sites. Imported automatic discovery routes must be from a sender site.
- If a PE router is configured to be in both sender sites and receiver sites, these guidelines apply:
    - Along with an import route target, you can optionally configure whether the route target is from a receiver or a sender site.
    - If a configuration is not provided, an imported automatic discovery route is treated as belonging to both the sender site set and the receiver site set.

To configure a route target for the MBGP MVPN routing instance, include the `route-target` statement:

```
route-target {
    export-target {
        target target-community;
        unicast;
    }
    import-target {
        target {
            target-value;
            receiver target-value;
            sender target-value;
        }
        unicast {
            receiver;
```

```
            sender;
    }
```

You can include this statement at the following hierarchy levels:

- `[edit routing-instances` *`routing-instance-name`* `protocols mvpn]`
- `[edit logical-systems` *`logical-system-name`* `routing-instances` *`routing-instance-name`* `protocols mvpn]`

The following sections describes how to configure the export target and the import target for an MBGP MVPN:

Configuring the Export Target for an MBGP MVPN

To configure an export target, include the `export-target` statement:

```
export-target {
    target target-community;
    unicast;
}
```

You can include this statement at the following hierarchy levels:

- `[edit routing-instances` *`routing-instance-name`* `protocols mvpn route-target]`
- `[edit logical-systems` *`logical-system-name`* `routing-instances` *`routing-instance-name`* `protocols mvpn route target]`

Configure the `target` option to specify the export target community. Configure the `unicast` option to use the same target community that has been specified for unicast.

Configuring the Import Target for an MBGP MVPN

To configure an import target, include the `import-target` statement:

```
import-target {
    target target-value {
        receiver;
        sender;
    }
    unicast {
        receiver;
        sender;
```

```
    }
}
```

You can include this statement at the following hierarchy levels:

- `[edit routing-instances routing-instance-name protocols mvpn route-target]`
- `[edit logical-systems logical-system-name routing-instances routing-instance-name protocols mvpn route-target]`

The following sections describe how to configure the import target and unicast parameters:

Configuring the Import Target Receiver and Sender for an MBGP MVPN

To configure the import target community, include the `target` statement and specify the target community. The target community must by in the format `target:x:y`. The `x` value is either an IP address or an AS number followed by an optional `L` to indicate a 4 byte AS number, and `y` is a number (for example, `target:123456L:100`)

```
target target-value {
    receiver;
    sender;
}
```

You can include this statement at the following hierarchy levels:

- `[edit routing-instances routing-instance-name protocols mvpn route-target import-target]`
- `[edit logical-systems logical-system-name routing-instances routing-instance-name protocols mvpn route-target import-target]`

You can specify the target community used when importing either receiver site sets or sender site sets by including one of the following statements:

- `receiver`—Specify the target community used when importing receiver site sets.
- `sender`—Specify the target community used when importing sender site sets.

Configuring the Import Target Unicast Parameters for an MBGP MVPN

To configure a unicast target community as the import target, include the `unicast` statement:

```
unicast {
    receiver;
```

```
    sender;
}
```

You can include this statement at the following hierarchy levels:

- `[edit routing-instances` *`routing-instance-name`* `protocols mvpn route-target import-target]`
- `[edit logical-systems` *`logical-system-name`* `routing-instances` *`routing-instance-name`* `protocols mvpn route-target import-target]`

You can specify the unicast target community used when importing either receiver site sets or sender site sets by including one of the following statements:

- `receiver`—Specify the unicast target community used when importing receiver site sets.
- `sender`—Specify the unicast target community used when importing sender site sets.

Limiting Routes to Be Advertised by an MVPN VRF Instance

If a hub-and-spoke deployment uses one VPN routing and forwarding (VRF) routing instance for unicast routing and a separate VRF for MVPN routing, you need to limit the PE routers at the hub site to advertise only IPv4 MVPN routes, only IPv6 MVPN routes, or both. This is necessary to prevent the multicast VRF instance from advertising unicast VPN routes to other PE routers.

NOTE: This configuration does not prevent the exportation of VPN routes to other VRF instances on the same router if the `auto-export` statement is included in the `[edit routing-options]` hierarchy.

To configure a VRF routing instance with the name `green` to advertise MVPN routes from both the `inet` and `inet6` address families, perform the following steps:

1. Configure the VRF routing instance to advertise IPv4 routes.

   ```
   user@host# set routing-instances green vrf-advertise-selective family inet-mvpn
   ```

2. Configure the VRF routing instance to advertise IPv6 routes.

   ```
   user@host# set routing-instances green vrf-advertise-selective family inet6-mvpn
   ```

After the configuration is committed, only the MVPN routes for the specified address families are advertised from the VRF instance to remote PE routers. To remove the restriction on routes being advertised, delete the `vrf-advertise-selective` statement.

NOTE: You cannot include the `vrf-advertise-selective` statement and the `no-vrf-advertise` statement in the same VRF configuration. However, if you configure the `vrf-advertise-selective` statement without any of its options, the router has the same behavior as if you configured the `no-vrf-advertise` statement. VPN routes are prevented from being advertised from a VRF routing instance to the remote PE routers.

SEE ALSO

family

inet-mvpn

inet6-mvpn

no-vrf-advertise

vrf-advertise-selective

Configuring Provider Tunnels in MVPNs

IN THIS SECTION

PIM Sparse Mode, PIM Dense Mode, Auto-RP, and BSR for MBGP MVPNs

You can configure PIM sparse mode, PIM dense mode, auto-RP, and bootstrap router (BSR) for MBGP MVPN networks:

- PIM sparse mode—Allows a router to use any unicast routing protocol and performs reverse-path forwarding (RPF) checks using the unicast routing table. PIM sparse mode includes an explicit join message, so routers determine where the interested receivers are and send join messages upstream to their neighbors, building trees from the receivers to the rendezvous point (RP).

- PIM dense mode—Allows a router to use any unicast routing protocol and performs reverse-path forwarding (RPF) checks using the unicast routing table. Packets are forwarded to all interfaces except the incoming interface. Unlike PIM sparse mode, where explicit joins are required for packets to be transmitted downstream, packets are flooded to all routers in the routing instance in PIM dense mode.

- Auto-RP—Uses PIM dense mode to propagate control messages and establish RP mapping. You can configure an auto-RP node in one of three different modes: discovery mode, announce mode, and mapping mode.

- BSR—Establishes RPs. A selected router in a network acts as a BSR, which selects a unique RP for different group ranges. BSR messages are flooded using a data tunnel between PE routers.

SEE ALSO

Example: Allowing MBGP MVPN Remote Sources

Example: Configuring a PIM-SSM Provider Tunnel for an MBGP MVPN

Configuring PIM Provider Tunnels for an MBGP MVPN

To configure a Protocol Independent Multicast (PIM) sparse mode provider tunnel for a multicast VPN, include the `pim-asm` statement:

```
pim-asm {
    group-address address;
}
```

You can include this statement at the following hierarchy levels:

- `[edit routing-instances` *`routing-instance-name`* `provider-tunnel]`

- `[edit logical-systems` *`logical-system-name`* `routing-instances` *`routing-instance-name`* `provider-tunnel]`

To complete the PIM sparse mode provider tunnel configuration, you also need to specify the group address using the `group-address` option. The source address for a PIM sparse mode provider tunnel is configured to be the loopback address of the loopback interface in the inet.0 routing table.

Configuring PIM-SSM GRE Selective Provider Tunnels

This topic describes how to configure a PIM-SSM GRE selective provider tunnel for an MBGP MVPN.

Creating a selective provider tunnel enables you to move high-rate traffic off the inclusive tunnel and deliver the multicast traffic only to receivers that request it. This improves bandwidth utilization.

To configure a PIM-SSM GRE selective provider tunnel for the 224.0.113.1/24 customer multicast group address, the 10.2.2.2/32 customer source address, and a virtual routing instance named `green`. Since we are using addresses outside the reserved SSM address range of 232.0.0.0/8, we must also include the `group-range` address with the `pim-ssm` option.

1. Configure the multicast group address range to be used for creating selective tunnels. The address prefix can be any valid nonreserved IPv4 multicast address range. Whether you configure a range of addresses or a single address, make sure that you configure enough group addresses for all the selective tunnels needed.

   ```
   user@host# set routing-instances green provider-tunnel selective group 224.0.113.1/24 source
   10.2.2.2/32 pim-ssm group-range 224.0.113.1/24
   ```

2. Configure the threshold rate in kilobits per second (Kbps) for triggering the creation of the selective tunnel. If you set the threshold rate to zero Kbps, the selective tunnel is created immediately, and the multicast traffic does not use an inclusive tunnel at all. Optionally, you can leave the threshold rate unconfigured and the result is the same as setting the threshold to zero.

   ```
   user@host# set routing-instances green provider-tunnel selective group 224.0.113.1/24 source
   10.2.2.2/32 threshold-rate 0
   ```

3. Configure the autonomous system number in the global routing options. This is required in MBGP MVPNs.

   ```
   user@host# set routing-options autonomous-system 100
   ```

When configuring PIM-SSM GRE selective provider tunnels, keep the following in mind:

- Aggregation of multiple customer multicast routes to a single PIM S-PMSI is not supported.
- Provider tunnel multicast group addresses must be IPv4 addresses, even in configurations in which the customer multicast group and source are IPv6 addresses.

SEE ALSO

Multicast VPN Terminology

pim-ssm

group-range

threshold-rate

Understanding Multicast Route Leaking for VRF and Virtual Router Instances

IN THIS SECTION

Service providers use Layer 3 VPNs to keep traffic separate and private for multiple customers. To separate a VPN's routes from routes in the public Internet or those in other VPNs, provider devices maintain separate routing tables (called VRF tables) for each VPN that connects to a customer edge device. Devices supporting customers or sites that belong to the VPN can access only the routes in the VRF tables for that VPN.

However, providers might need to share services with more than one customer or site while keeping general services private among their customers. Providers can achieve this by making some routes available in the routing tables for particular customer VRF or virtual router instances. This practice is called *route leaking*, which enables a device to share route information from one configured VRF or virtual router routing instance to another using internal route export and import policies.

Providers might also want to use route leaking for multicast services such as IPTV and other streaming media services.

Static Multicast VRF Route Leaking Implementation on Switches

On Junos OS switches, the multicast VRF route leaking implementation enables you to statically share multicast routes from a Layer 3 VPN routing instance running a multicast protocol such as Protocol Independent Multicast (PIM) with customer virtual router or VRF instances. You can leak only static multicast routes with a prefix length of /32. Therefore, routes are shared for Internet Group Management Protocol (IGMP) groups and not for a specific source. All Layer 2 and Layer 3 interfaces must have IGMP version 3 enabled. No other version of IGMP is supported.

Additionally, you must configure an integrated routing and bridging (IRB) interface for each Layer 3 interface. To ensure that the multicast static routes are present in the routing instance that is running multicast, use IGMP to add the routes to each IRB interface configured in the routing instance. Include the `group` *`multicast-group-address`* `source` *`ip-address`* statements at the `[edit protocols igmp interface` *`irb-interface-name`* `static]` hierarchy level.

NOTE: For multicast route leaking to work robustly, you must also configure Protocol-Independent Multicast (PIM) on each IRB interface included in the virtual router or VRF instance.

- For example, to add the IRB interface, `irb.1023`, to a routing instance named `cust-11` and enable PIM on the IRB interface:

```
user@switch# set routing-instances cust-11 interface irb.1023
user@switch# set routing-instances cust-11 protocols pim interface irb.1023
```

This implementation also requires you to enable IGMP snooping on all customer interfaces receiving multicast traffic. Use the `multicast-router-interface` statement to configure each customer interface to face the multicast routing instance. You must also add each multicast group to each customer interface by including the `group` *`multicast-group-address`* statement at the `[edit protocols igmp-snooping vlan` *`vlan-id`* `interface` *`interface-name`*`]` hierarchy level. The customer VRF or virtual router instances are not required to have a multicast protocol such as PIM configured.

To enable multicast route leaking, configure static multicast routes in a customer VRF or virtual router instance for each configured multicast group and point each route to the routing table for the multicast routing instance. To configure, include the `static` `route` *`destination-prefix`*`/32` `next-table` *`instance-name`*`.inet.0` group of statements at the `[edit routing-instances` *`routing-instance-name`* `routing-options]` hierarchy level.

For example, to leak multicast route 233.252.0.0/32 to a customer routing instance named `cust-11`:

```
user@switch# set routing-instances cust-11 routing-options route 233.252.0.0/32 next table
HQ.inet.0
```

In this example, the static multicast route is configured on the customer routing instance and points to the routing table for the multicast routing instance, `HQ`. This configuration ensures that multicast traffic is forwarded.

RELATED DOCUMENTATION

Routing Instances in Layer 3 VPNs | **19**

igmp

igmp-snooping

CHAPTER 5

Full-Mesh, Hub-and-Spoke, and Overlapping VPNs

IN THIS CHAPTER

Full Mesh VPNs

IN THIS SECTION

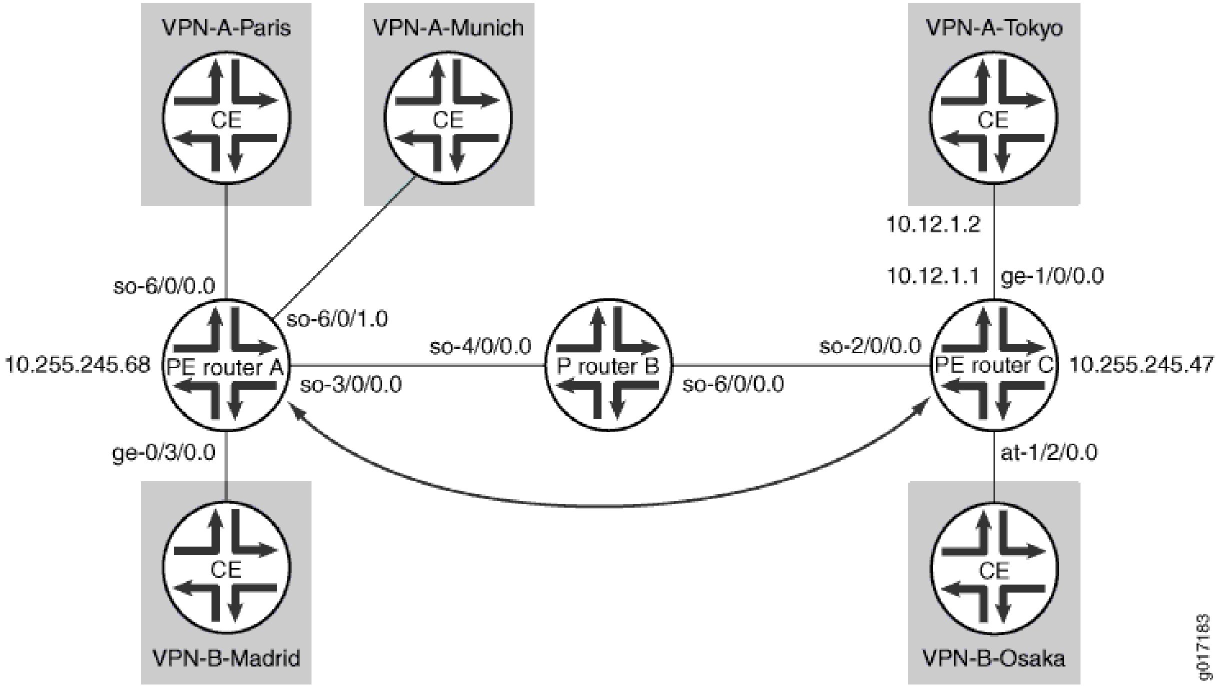
Configuring a Simple Full-Mesh VPN Topology

IN THIS SECTION

This example shows how to set up a simple full-mesh service provider VPN configuration, which consists of the following components (see Figure 63 on page 796):

- Two separate VPNs (VPN-A and VPN-B)
- Two provider edge (PE) routers, both of which service VPN-A and VPN-B
- RSVP as the signaling protocol
- One RSVP label-switched path (LSP) that tunnels between the two PE routers through one provider (P) router

Figure 63: Example of a Simple VPN Topology



In this configuration, route distribution in VPN A from Router VPN-A-Paris to Router VPN-A-Tokyo occurs as follows:

1. The customer edge (CE) router VPN-A-Paris announces routes to the PE router Router A.
2. Router A installs the received announced routes into its VPN routing and forwarding (VRF) table, VPN-A.inet.0.
3. Router A creates an MPLS label for the interface between it and Router VPN-A-Paris.
4. Router A checks its VRF export policy.

5. Router A converts the Internet Protocol version 4 (IPv4) routes from Router VPN-A-Paris into VPN IPv4 format using its route distinguisher and announces these routes to PE Router C over the IBGP between the two PE routers.
6. Router C checks its VRF import policy and installs all routes that match the policy into its bgp.l3vpn.0 routing table. (Any routes that do not match are discarded.)
7. Router C checks its VRF import policy and installs all routes that match into its VPN-A.inet.0 routing table. The routes are installed in IPv4 format.
8. Router C announces its routes to the CE router Router VPN-A-Tokyo, which installs them into its primary routing table. (For routing platforms running Junos OS, the primary routing table is inet.0.)
9. Router C uses the LSP between it and Router A to route all packets from Router VPN-A-Tokyo that are destined for Router VPN-A-Paris.

The final section in this example consolidates the statements needed to configure VPN functionality on each of the service P routers shown in Figure 63 on page 796.

NOTE: In this example, a private autonomous system (AS) number is used for the route distinguisher and the route target. This number is used for illustration only. When you are configuring VPNs, you should use an assigned AS number.

The following sections explain how to configure the VPN functionality on the PE and P routers. The CE routers have no information about the VPN, so you configure them normally.

Enabling an IGP on the PE and P Routers

To allow the PE and P routers to exchange routing information among themselves, you must configure an interior gateway protocol (IGP) on all these routers or you must configure static routes. You configure the IGP on the primary instance of the routing protocol process (rpd) (that is, at the `[edit protocols]` hierarchy level), not within the VPN routing instance (that is, not at the `[edit routing-instances]` hierarchy level).

You configure the IGP in the standard way. This configuration example does not include this portion of the configuration.

Enabling RSVP and MPLS on the P Router

On the P router, Router B, you must configure RSVP and MPLS because this router exists on the MPLS LSP path between the two PE routers, Router A and Router C:

```
[edit]
protocols {
    rsvp {
        interface so-4/0/0.0;
        interface so-6/0/0.0;
    }
    mpls {
        interface so-4/0/0.0;
        interface so-6/0/0.0;
    }
}
```

Configuring the MPLS LSP Tunnel Between the PE Routers

In this configuration example, RSVP is used for VPN signaling. Therefore, in addition to configuring RSVP, you must enable traffic engineering support in an IGP and you must create an MPLS LSP to tunnel the VPN traffic.

On PE Router A, enable RSVP and configure one end of the MPLS LSP tunnel. In this example, traffic engineering support is enabled for OSPF. When configuring the MPLS LSP, include `interface` statements for all interfaces participating in MPLS, including the interfaces to the PE and CE routers. The statements for the interfaces between the PE and CE routers are needed so that the PE router can create an MPLS label for the private interface. In this example, the first `interface` statement configures MPLS on the interface connected to the LSP, and the remaining three configure MPLS on the interfaces that connect the PE router to the CE routers.

```
[edit]
protocols {
    rsvp {
        interface so-3/0/0.0;
    }
    mpls {
        label-switched-path RouterA-to-RouterC {
            to 10.255.245.47;
        }
        interface so-3/0/0.0;
```

```
        interface so-6/0/0.0;
        interface so-6/0/1.0;
        interface ge-0/3/0.0;
    }
    ospf {
        traffic-engineering;
        area 0.0.0.0 {
            interface so-3/0/0.0;
        }
    }
}
```

On PE Router C, enable RSVP and configure the other end of the MPLS LSP tunnel. Again, traffic engineering support is enabled for OSPF, and you configure MPLS on the interfaces to the LSP and the CE routers.

```
[edit]
protocols {
    rsvp {
        interface so-2/0/0.0;
    }
    mpls {
        label-switched-path RouterC-to-RouterA {
            to 10.255.245.68;
        }
        interface so-2/0/0.0;
        interface ge-1/0/0.0;
        interface at-1/2/0.0;
    }
    ospf {
        traffic-engineering;
        area 0.0.0.0 {
            interface so-2/0/0.0;
        }
    }
}
```

Configuring IBGP on the PE Routers

On the PE routers, configure an IBGP session with the following properties:

- VPN family—To indicate that the IBGP session is for the VPN, include the `family inet-vpn` statement.

- Loopback address—Include the `local-address` statement, specifying the local PE router's loopback address. The IBGP session for VPNs runs through the loopback address. You must also configure the `lo0` interface at the `[edit interfaces]` hierarchy level. The example does not include this part of the router's configuration.
- Neighbor address—Include the `neighbor` statement, specifying the IP address of the neighboring PE router, which is its loopback (`lo0`) address.

On PE Router A, configure IBGP:

```
[edit]
protocols {
    bgp {
        group PE-RouterA-to-PE-RouterC {
            type internal;
            local-address 10.255.245.68;
            family inet-vpn {
                unicast;
            }
            neighbor 10.255.245.47;
        }
    }
}
```

On PE Router C, configure IBGP:

```
[edit]
protocols {
    bgp {
        group PE-RouterC-to-PE-RouterA {
            type internal;
            local-address 10.255.245.47;
            family inet-vpn {
                unicast;
            }
            neighbor 10.255.245.68;
        }
    }
}
```

Configuring Routing Instances for VPNs on the PE Routers

Both PE routers service VPN-A and VPN-B, so you must configure two routing instances on each router, one for each VPN. For each VPN, you must define the following in the routing instance:

- Route distinguisher, which must be unique for each routing instance on the PE router.
- It is used to distinguish the addresses in one VPN from those in another VPN.
- Instance type of `vrf`, which creates the VRF table on the PE router.
- Interfaces connected to the CE routers.
- VRF import and export policies, which must be the same on each PE router that services the same VPN. Unless an import policy contains only a `then reject` statement, it must include reference to a community. Otherwise, when you try to commit the configuration, the commit fails.

  NOTE: In this example, a private AS number is used for the route distinguisher. This number is used for illustration only. When you are configuring VPNs, you should use an assigned AS number.

- Routing between the PE and CE routers, which is required for the PE router to distribute VPN-related routes to and from connected CE routers. You can configure a routing protocol—BGP, OSPF, or RIP—or you can configure static routing.

On PE Router A, configure the following routing instance for VPN-A. In this example, Router A uses static routes to distribute routes to and from the two CE routers to which it is connected.

```
[edit]
routing-instance {
    VPN-A-Paris-Munich {
        instance-type vrf;
        interface so-6/0/0.0;
        interface so-6/0/1.0;
        route-distinguisher 65535:0;
        vrf-import VPN-A-import;
        vrf-export VPN-A-export;
        routing-options {
            static {
                route 172.16.0.0/16 next-hop so-6/0/0.0;
                route 172.17.0.0/16 next-hop so-6/0/1.0;
            }
        }
```

```
    }
}
```

On PE Router C, configure the following routing instance for VPN-A. In this example, Router C uses BGP to distribute routes to and from the CE router to which it is connected.

```
[edit]
routing-instance {
    VPN-A-Tokyo {
        instance-type vrf;
        interface ge-1/0/0.0;
        route-distinguisher 65535:1;
        vrf-import VPN-A-import;
        vrf-export VPN-A-export;
        protocols {
            bgp {
                group VPN-A-Site2 {
                    peer-as 1;
                    neighbor 10.12.1.2;
                }
            }
        }
    }
}
```

On PE Router A, configure the following routing instance for VPN-B. In this example, Router A uses OSPF to distribute routes to and from the CE router to which it is connected.

```
[edit]
policy-options {
    policy-statement bgp-to-ospf {
        from {
            protocol bgp;
            route-filter 192.168.1.0/24 orlonger;
        }
        then accept;
    }
}
routing-instance {
    VPN-B-Madrid {
        instance-type vrf;
```

```
        interface ge-0/3/0.0;
        route-distinguisher 65535:2;
        vrf-import VPN-B-import;
        vrf-export VPN-B-export;
        protocols {
            ospf {
                export bgp-to-ospf;
                area 0.0.0.0 {
                    interface ge-0/3/0;
                }
            }
        }
    }
}
```

On PE Router C, configure the following routing instance for VPN-B. In this example, Router C uses RIP to distribute routes to and from the CE router to which it is connected.

```
[edit]
policy-options {
    policy-statement bgp-to-rip {
        from {
            protocol bgp;
            route-filter 192.168.2.0/24 orlonger;
        }
        then accept;
    }
}
routing-instance {
    VPN-B-Osaka {
        instance-type vrf;
        interface at-1/2/0.0;
        route-distinguisher 65535:3;
        vrf-import VPN-B-import;
        vrf-export VPN-B-export;
        protocols {
            rip {
                group PE-C-to-VPN-B {
                    export bgp-to-rip;
                    neighbor at-1/2/0;
                }
            }
```

```
        }
    }
}
```

Configuring VPN Policy on the PE Routers

Configure the VPN import and export policies on each PE router so that the appropriate routes are installed in the PE router’s VRF tables. The VRF table is used to forward packets within a VPN. For VPN-A, the VRF table is VPN-A.inet.0, and for VPN-B it is VPN-B.inet.0.

In the VPN policy, you also configure VPN target communities.

In the following example, a private AS number is used for the route target. This number is used for illustration only. When you are configuring VPNs, you should use an assigned AS number. The policy qualifiers shown in this example are only those needed for the VPN to function. You can configure additional qualifiers, as needed, for any policies that you configure.

On PE Router A, configure the following VPN import and export policies:

```
[edit]
policy-options {
    policy-statement VPN-A-import {
        term a {
            from {
                protocol bgp;
                community VPN-A;
            }
            then accept;
        }
        term b {
            then reject;
        }
    }
    policy-statement VPN-A-export {
        term a {
            from protocol static;
            then {
                community add VPN-A;
                accept;
            }
        }
        term b {
            then reject;
```

```
        }
    }
    policy-statement VPN-B-import {
        term a {
            from {
                protocol bgp;
                community VPN-B;
            }
            then accept;
        }
        term b {
            then reject;
        }
    }
    policy-statement VPN-B-export {
        term a {
            from protocol ospf;
            then {
                community add VPN-B;
                accept;
            }
        }
        term b {
            then reject;
        }
    }
    community VPN-A members target:65535:4;
    community VPN-B members target:65535:5;
}
```

On PE Router C, configure the following VPN import and export policies:

```
[edit]
policy-options {
    policy-statement VPN-A-import {
        term a {
            from {
                protocol bgp;
                community VPN-A;
            }
            then accept;
        }
```

```
        term b {
            then reject;
        }
    }
    policy-statement VPN-A-export {
        term a {
            from protocol bgp;
            then {
                community add VPN-A;
                accept;
            }
        }
        term b {
            then reject;
        }
    }
    policy-statement VPN-B-import {
        term a {
            from {
                protocol bgp;
                community VPN-B;
            }
            then accept;
        }
        term b {
            then reject;
        }
    }
    policy-statement VPN-B-export {
        term a {
            from protocol rip;
            then {
                community add VPN-B;
                accept;
            }
        }
        term b {
            then reject;
        }
    }
    community VPN-A members target:65535:4;
```

```
    community VPN-B members target:65535:5;
}
```

To apply the VPN policies on the routers, include the `vrf-export` and `vrf-import` statements when you configure the routing instance. For both VPNs, the VRF import and export policies handle the route distribution across the IBGP session running between the PE routers.

To apply the VPN policies on PE Router A, include the following statements:

```
[edit]
routing-instance {
    VPN-A-Paris-Munich {
        vrf-import VPN-A-import;
        vrf-export VPN-A-export;
    }
    VPN-B-Madrid {
        vrf-import VPN-B-import;
        vrf-export VPN-B-export;
    }
}
```

To apply the VPN policies on PE Router C, include the following statements:

```
[edit]
routing-instance {
    VPN-A-Tokyo {
        vrf-import VPN-A-import;
        vrf-export VPN-A-export;
    }
    VPN-B-Osaka {
        vrf-import VPN-B-import;
        vrf-export VPN-B-export;
    }
}
```

Simple VPN Configuration Summarized by Router

Router A (PE Router)

Routing Instance for VPN-A

```
routing-instance {
    VPN-A-Paris-Munich {
        instance-type vrf;
        interface so-6/0/0.0;
        interface so-6/0/1.0;
        route-distinguisher 65535:0;
        vrf-import VPN-A-import;
        vrf-export VPN-A-export;
    }
}
```

Instance Routing Protocol

```
routing-options {
    static {
        route 172.16.0.0/16 next-hop so-6/0/0.0;
        route 172.17.0.0/16 next-hop so-6/0/1.0;
    }
}
```

Routing Instance for VPN-B

```
routing-instance {
    VPN-B-Madrid {
        instance-type vrf;
        interface ge-0/3/0.0;
        route-distinguisher 65535:2;
        vrf-import VPN-B-import;
        vrf-export VPN-B-export;
    }
}
```

Instance Routing Protocol

```
protocols {
    ospf {
        area 0.0.0.0 {
            interface ge-0/3/0;
        }
    }
}
```

Primary Protocol Instance

```
protocols {
}
```

Enable RSVP

```
rsvp {
    interface so-3/0/0.0;
}
```

Configure an MPLS LSP

```
mpls {
    label-switched-path RouterA-to-RouterC {
        to 10.255.245.47;
    }
    interface so-3/0/0.0;
    interface so-6/0/0.0;
    interface so-6/0/1.0;
    interface ge-0/3/0.0;
}
```

Configure IBGP

```
bgp {
    group PE-RouterA-to-PE-RouterC {
        type internal;
        local-address 10.255.245.68;
```

```
        family inet-vpn {
            unicast;
        }
        neighbor 10.255.245.47;
    }
}
```

Configure OSPF for Traffic Engineering Support

```
ospf {
    traffic-engineering;
    area 0.0.0.0 {
        interface so-3/0/0.0;
    }
}
```

Configure VPN Policy

```
policy-options {
    policy-statement VPN-A-import {
        term a {
            from {
                protocol bgp;
                community VPN-A;
            }
            then accept;
        }
        term b {
            then reject;
        }
    }
    policy-statement VPN-A-export {
        term a {
            from protocol static;
            then {
                community add VPN-A;
                accept;
            }
        }
        term b {
            then reject;
```

```
            }
        }
        policy-statement VPN-B-import {
            term a {
                from {
                    protocol bgp;
                    community VPN-B;
                }
                then accept;
            }
            term b {
                then reject;
            }
        }
        policy-statement VPN-B-export {
            term a {
                from protocol ospf;
                then {
                    community add VPN-B;
                    accept;
                }
            }
            term b {
                then reject;
            }
        }
        community VPN-A members target:65535:4;
        community VPN-B members target:65535:5;
    }
```

Router B (P Router)

Primary Protocol Instance

```
protocols {
}
```

Enable RSVP

```
rsvp {
    interface so-4/0/0.0;
    interface so-6/0/0.0;
}
```

Enable MPLS

```
mpls {
    interface so-4/0/0.0;
    interface so-6/0/0.0;
}
```

Router C (PE Router)

Routing Instance for VPN-A

```
routing-instance {
    VPN-A-Tokyo {
        instance-type vrf;
        interface ge-1/0/0.0;
        route-distinguisher 65535:1;
        vrf-import VPN-A-import;
        vrf-export VPN-A-export;
    }
}
```

Instance Routing Protocol

```
protocols {
    bgp {
        group VPN-A-Site2 {
            peer-as 1;
            neighbor 10.12.1.2;
        }
    }
}
```

Routing Instance for VPN-B

```
VPN-B-Osaka {
    instance-type vrf;
    interface at-1/2/0.0;
    route-distinguisher 65535:3;
    vrf-import VPN-B-import;
    vrf-export VPN-B-export;
}
```

Instance Routing Protocol

```
protocols {
    rip {
        group PE-C-to-VPN-B {
            neighbor at-1/2/0;
        }
    }
}
```

Primary Protocol Instance

```
protocols {
}
```

Enable RSVP

```
rsvp {
    interface so-2/0/0.0;
}
```

Configure an MPLS LSP

```
mpls {
    label-switched-path RouterC-to-RouterA {
        to 10.255.245.68;
    }
    interface so-2/0/0.0;
    interface ge-1/0/0.0;
```

```
    interface at-1/2/0.0;
}
```

Configure IBGP

```
bgp {
    group PE-RouterC-to-PE-RouterA {
        type internal;
        local-address 10.255.245.47;
        family inet-vpn {
            unicast;
        }
        neighbor 10.255.245.68;
    }
}
```

Configure OSPF for Traffic Engineering Support

```
ospf {
    traffic-engineering;
    area 0.0.0.0 {
        interface so-2/0/0.0;
    }
}
```

Configure VPN Policy

```
policy-options {
    policy-statement VPN-A-import {
        term a {
            from {
                protocol bgp;
                community VPN-A;
            }
            then accept;
        }
        term b {
            then reject;
        }
    }
    policy-statement VPN-A-export {
```

```
        term a {
            from protocol bgp;
            then {
                community add VPN-A;
                accept;
            }
        }
        term b {
            then reject;
        }
    }
    policy-statement VPN-B-import {
        term a {
            from {
                protocol bgp;
                community VPN-B;
            }
            then accept;
        }
        term b {
            then reject;
        }
    }
    policy-statement VPN-B-export {
        term a {
            from protocol rip;
            then {
                community add VPN-B;
                accept;
            }
        }
        term b {
            then reject;
        }
    }
    community VPN-A members target:65535:4;
    community VPN-B members target:65535:5;
}
```

Configuring a Full-Mesh VPN Topology with Route Reflectors

This example is a variation of the full-mesh VPN topology example (described in "Configuring a Simple Full-Mesh VPN Topology" on page 795) in which one of the PE routers is a BGP route reflector. In this variation, Router C in "Configuring a Simple Full-Mesh VPN Topology" on page 795 is a route reflector. The only change to its configuration is that you need to include the `cluster` statement when configuring the BGP group:

```
[edit]
protocols {
    bgp {
        group PE-RouterC-to-PE-RouterA {
            type internal;
            local-address 10.255.245.47;
            family inet-vpn {
                unicast;
            }
            neighbor 10.255.245.68;
            cluster 4.3.2.1;
        }
    }
}
```

Hub-and-Spoke VPNs

IN THIS SECTION

Configuring Hub-and-Spoke VPN Topologies: One Interface

IN THIS SECTION

Use a one-interface configuration to advertise a default route from a hub or hubs.

Figure 64: Example of a Hub-and-Spoke VPN Topology with One Interface

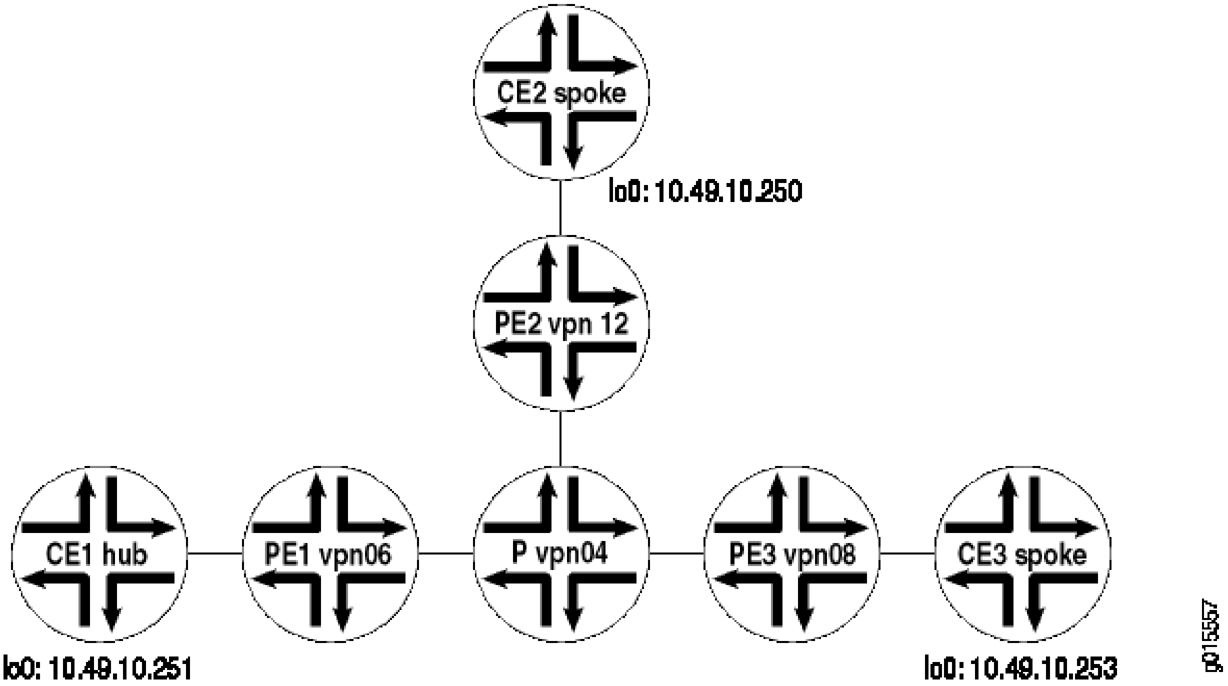


Figure 64 on page 817 illustrates a Layer 3 VPN hub-and-spoke application where there is only one interface between the hub CE (CE1) and the hub PE (PE1). This is the recommended way of configuring hub-and-spoke topologies.

In this configuration, a default route is advertised from the hub to the spokes. If more specific spoke CE routes need to be exchanged between spoke CE routers, then two interfaces are needed between the hub CE and hub PE. See "Configuring Hub-and-Spoke VPN Topologies: Two Interfaces" on page 832 for a two-interface example.

In this configuration example, spoke route distribution is as follows:

1. Spoke CE2 advertises its routes to spoke PE2.
2. Spoke PE2 installs routes from CE2 into its VPN routing and forwarding (VRF) table.
3. Spoke PE2 checks its VRF export policy, adds the route target community, and announces the routes to hub PE1.
4. Hub PE1 checks its VRF import policy and installs routes that match the import policy into table bgp.l3vpn.0.
5. Hub PE1 installs routes from table bgp.l3vpn.0 into the hub VRF table.
6. Hub PE1 announces routes from the hub VRF table to the hub CE1.

In this configuration example, default route distribution is as follows:

1. Hub CE1 announces a default route to hub PE1.
2. Hub PE1 installs the default route into the hub VRF table.
3. Hub PE1 checks its VRF export policy, adds the route target community and announces the default route to spoke PE2 and PE3.
4. Spoke PE2 and PE3 check their VRF import policy and install the default route into table bgp.l3vpn.0.
5. Spoke PE2 and PE3 install the routes from table bgp.l3vpn.0 into their spoke VRF tables.
6. Spoke PE2 and PE3 announce the default route from the spoke VRF table to spoke CE2 and CE3.

The following sections describe how to configure a hub-and-spoke topology with one interface based on the topology illustrated in Figure 64 on page 817:

Configuring Hub CE1

Configure hub CE1 as follows:

```
[edit routing-options]
static {
    route 0.0.0.0/0 discard;
}
autonomous-system 100;
[edit protocols]
bgp {
    group hub {
        type external;
        export default;
```

```
        peer-as 200;
        neighbor 10.49.4.1;
    }
}
[edit policy-options]
policy-statement default {
    term 1 {
        from {
            protocol static;
            route-filter 0.0.0.0/0 exact;
        }
        then accept;
    }
    term 2 {
        then reject;
    }
}
```

Configuring Hub PE1

Configure hub PE1 as follows:

```
[edit]
routing-instances {
    hub {
        instance-type vrf;
        interface t3-0/0/0;
        route-distinguisher 10.255.14.176:2;
        vrf-target {
            import target:200:100;
            export target:200:101;
        }
        protocols {
            bgp {
                group hub {
                    type external;
                    peer-as 100;
                    as-override;
                    neighbor 10.49.4.2;
                }
            }
```

```
        }
    }
}
```

Configuring the P Router

Configure the P Router as follows:

```
[edit]
interfaces {
    t3-0/1/1 {
        unit 0 {
            family inet {
                address 10.49.2.1/30;
            }
            family mpls;
        }
    }
    t3-0/1/3 {
        unit 0 {
            family inet {
                address 10.49.0.2/30;
            }
            family mpls;
        }
    }
    t1-0/2/0 {
        unit 0 {
            family inet {
                address 10.49.1.2/30;
            }
            family mpls;
        }
    }
}
[edit]
protocols {
    ospf {
        area 0.0.0.0 {
            interface t3-0/1/3.0;
            interface t1-0/2/0.0;
```

```
            interface t3-0/1/1.0;
            interface lo0.0 {
                passive;
            }
        }
    }
    ldp {
        interface t3-0/1/1.0;
        interface t3-0/1/3.0;
        interface t1-0/2/0.0;
    }
}
```

Configuring Spoke PE2

Configure spoke PE2 as follows:

```
[edit]
interfaces {
    t3-0/0/0 {
        unit 0 {
            family inet {
                address 10.49.0.1/30;
            }
            family mpls;
        }
    }
    t1-0/1/2 {
        unit 0 {
            family inet {
                address 10.49.3.1/30;
            }
        }
    }
}
[edit protocols]
bgp {
    group ibgp {
        type internal;
        local-address 10.255.14.182;
        peer-as 200;
```

```
        neighbor 10.255.14.176 {
            family inet-vpn {
                unicast;
            }
        }
    }
}
ospf {
    area 0.0.0.0 {
        interface t3-0/0/0.0;
        interface lo0.0 {
            passive;
        }
    }
}
ldp {
    interface t3-0/0/0.0;
}
[edit]
routing-instances {
    spoke {
        instance-type vrf;
        interface t1-0/1/2.0;
        route-distinguisher 10.255.14.182:20;
        vrf-target {
            import target:200:101;
            export target:200:100;
        }
        protocols {
            bgp {
                group spoke {
                    type external;
                    peer-as 100;
                    as-override;
                    neighbor 10.49.3.2;
                }
            }
        }
    }
}
```

Configuring Spoke PE3

Configure spoke PE3 as follows:

```
[edit]
interfaces {
    t3-0/0/0 {
        unit 0 {
            family inet {
                address 10.49.6.1/30;
            }
        }
    }
    t3-0/0/1 {
        unit 0 {
            family inet {
                address 10.49.2.2/30;
            }
            family mpls;
        }
    }
}
[edit protocols}
bgp {
    group ibgp {
        type internal;
        local-address 10.255.14.178;
        peer-as 200;
        neighbor 10.255.14.176 {
            family inet-vpn {
                unicast;
            }
        }
    }
}
ospf {
    area 0.0.0.0 {
        interface t3-0/0/1.0;
        interface lo0.0 {
            passive;
        }
    }
```

```
}
ldp {
    interface t3-0/0/1.0;
}
[edit]
routing-instances {
    spoke {
        instance-type vrf;
        interface t3-0/0/0.0;
        route-distinguisher 10.255.14.178:30;
        vrf-target {
            import target:200:101;
            export target:200:100;
        }
        protocols {
            bgp {
                group spoke {
                    type external;
                    peer-as 100;
                    as-override;
                    neighbor 10.49.6.2;
                }
            }
        }
    }
}
```

Configuring Spoke CE2

Configure spoke CE2 as follows:

```
[edit routing-options]
autonomous-system 100;
{edit protocols]
bgp {
    group spoke {
        type external;
        export loopback;
        peer-as 200;
        neighbor 10.49.3.1;
```

```
    }
}
```

Configuring Spoke CE3

Configure spoke CE3 as follows:

```
[edit routing-options]
autonomous-system 100;
[edit protocols]
bgp {
    group spoke {
        type external;
        export loopback;
        peer-as 200;
        neighbor 10.49.6.1;
    }
}
```

In this configuration example, traffic forwarding is as follows between spoke CE2 and hub CE1:

1. Spoke CE2 forwards traffic using the default route learned from spoke PE2 through BGP.

```
      0.0.0.0/0          *[BGP/170] 02:24:15, localpref 100
                            AS path: 200 200 I
                          > to 10.49.3.1 via t1-3/0/1.0
```

2. Spoke PE2 performs a route lookup in the spoke VRF table and forwards the traffic to hub PE1 (through the P router—PE2 pushes two labels) using the default route learned through BGP.

```
      0.0.0.0/0          *[BGP/170] 01:35:45, localpref 100, from 10.255.14.176
                            AS path: 100 I
                          > via t3-0/0/1.0, Push 100336, Push 100224(top)
```

3. Hub PE1 does a route lookup in the mpls.0 table for the VPN label `100336`.

```
      100336             *[VPN/170] 01:37:03
                          > to 10.49.4.2 via t3-0/0/0.0, Pop
```

4. Hub PE1 forwards the traffic out the interface `t3-0/0/0.0` to hub CE1.

In this configuration example, traffic forwarding is as follows between hub CE1 and spoke CE2:

1. Hub CE1 forwards traffic to the hub PE1 using the route learned through BGP.

```
10.49.10.250/32    *[BGP/170] 02:28:46, localpref 100
                      AS path: 200 200 I
                    > to 10.49.4.1 via t3-3/1/0.0
```

2. Hub PE1 does a route lookup in the hub VRF table and forwards the traffic to spoke PE2 (through the P router—PE1 pushes two labels).

```
10.49.10.250/32    *[BGP/170] 01:41:05, localpref 100, from 10.255.14.182
                      AS path: 100 I
                    > via t1-0/1/0.0, Push 100352, Push 100208(top)
```

3. Spoke PE2 does a route lookup in the mpls.0 table for the VPN label `100352`.

```
100352             *[VPN/170] 02:31:39
                    > to 10.49.3.2 via t1-0/1/2.0, Pop
```

4. Spoke PE2 forwards the traffic out the interface `t1-0/1/2.0` to spoke CE2.

In this configuration example, traffic forwarding is as follows between spoke CE2 and spoke CE3:

1. Spoke CE2 forwards traffic using the default route learned from spoke PE2 through BGP.

```
0.0.0.0/0          *[BGP/170] 02:24:15, localpref 100
                      AS path: 200 200 I
                    > to 10.49.3.1 via t1-3/0/1.0
```

2. Spoke PE2 does a route lookup in the spoke VRF table and forwards the traffic to hub PE1 (through the P router—PE2 pushes two labels) using the default route learned through BGP.

```
0.0.0.0/0          *[BGP/170] 01:35:45, localpref 100, from 10.255.14.176
                      AS path: 100 I
                    > via t3-0/0/1.0, Push 100336, Push 100224(top)
```

3. Hub PE1 does a route lookup in the mpls.0 table for the VPN label `100336`.

```
100336             *[VPN/170] 01:37:03
                    > to 10.49.4.2 via t3-0/0/0.0, Pop
```

4. Hub PE1 forwards the traffic out the interface `t3-0/0/0.0` to the hub CE1.

5. Hub CE1 forwards the traffic to hub PE1 using the router learned through BGP.

```
10.49.10.253/32    *[BGP/170] 02:40:03, localpref 100
                      AS path: 200 200 I
                    > to 10.49.4.1 via t3-3/1/0.0
```

6. Hub PE1 does a route lookup in the hub VRF table and forwards the traffic to spoke PE3 (through the P router—PE1 pushes two labels).

```
10.49.10.253/32    *[BGP/170] 01:41:05, localpref 100, from 10.255.14.178
                      AS path: 100 I
                    > via t1-0/1/0.0, Push 100128, Push 100192(top)
```

7. Spoke PE3 does a route lookup in the mpls.0 table for VPN label `100128`.

```
100128             *[VPN/170] 02:41:30
                    > to 10.49.6.2 via t3-0/0/0.0, Pop
```

8. Spoke PE3 forwards the traffic out the interface `t3-0/0/0.0` to spoke CE3.

If egress features are needed on the hub PE that require an IP forwarding lookup on the hub VRF routing table, see "Enabling Egress Features on the Hub PE Router" on page 827.

Enabling Egress Features on the Hub PE Router

This example is provided in conjunction with "Configuring Hub-and-Spoke VPN Topologies: One Interface" on page 816. This example also uses the topology illustrated in Figure 64 on page 817.

If egress features are needed on the hub PE that require an IP forwarding lookup on the hub VRF routing table, the configuration detailed in "Configuring Hub-and-Spoke VPN Topologies: One Interface" on page 816 will not work. Applying the `vrf-table-label` statement on the hub routing instance forces traffic from a remote spoke PE to be forwarded to the hub PE and forces an IP lookup to be performed.

Because specific spoke routes are in the hub VRF table, traffic will be forwarded to a spoke PE without going through the hub CE.

The hub PE advertises the default route as follows, using VPN label 1028:

```
hub.inet.0: 7 destinations, 7 routes (7 active, 0 holddown, 0 hidden)
* 0.0.0.0/0 (1 entry, 1 announced)
 BGP group ibgp type Internal
     Route Distinguisher: 10.255.14.176:2
     VPN Label: 1028
     Nexthop: Self
     Localpref: 100
     AS path: 100 I
     Communities: target:200:101
```

Incoming traffic is forwarded using VPN label 1028. The mpls.0 table shows that an IP lookup in the table hub.inet.0 is required:

```
1028               *[VPN/0] 00:00:27
                      to table hub.inet.0, Pop
```

However, the hub VRF table hub.inet.0 contains specific spoke routes:

```
10.49.10.250/32    *[BGP/170] 00:00:05, localpref 100, from 10.255.14.182
                      AS path: 100 I
                    > via t1-0/1/0.0, Push 100352, Push 100208(top)
10.49.10.253/32    *[BGP/170] 00:00:05, localpref 100, from 10.255.14.178
                      AS path: 100 I
                    > via t1-0/1/0.0, Push 100128, Push 100192(top)
```

Because of this, traffic is forwarded directly to the spoke PEs without going through the hub CE. To prevent this, you must configure a secondary routing instance for downstream traffic in the hub PE1.

Configuring Hub PE1

Configure hub PE1 as follows:

```
[edit]
routing-instances {
    hub {
```

```
        instance-type vrf;
        interface t3-0/0/0.0;
        route-distinguisher 10.255.14.176:2;
        vrf-target {
            import target:200:100;
            export target:200:101;
        }
        no-vrf-advertise;
        routing-options {
            auto-export;
        }
        protocols {
            bgp {
                group hub {
                    type external;
                    peer-as 100;
                    as-override;
                    neighbor 10.49.4.2;
                }
            }
        }
    }
    hub-downstream {
        instance-type vrf;
        route-guisher 10.255.14.176:3;
        vrf-target target:200:101;
        vrf-table-label;
        routing-options {
            auto-export;
        }
    }
}
```

When the `no-vrf-advertise` statement is used at the `[edit routing-instances hub]` hierarchy level, no routing table groups or VRF export policies are required. The `no-vrf-advertise` statement configures the hub PE not to advertise VPN routes from the primary routing-instance `hub`. These routes are instead advertised from the secondary routing instance `hub_downstream`. See Junos OS Routing Protocols Library for more information about the `no-vrf-advertise` statement.

The `auto-export` statement at the `[edit routing-instances hub-downstream routing-options]` hierarchy level identifies routes exported from the hub instance to the hub-downstream instance by looking at the route targets defined for each routing instance. See Junos OS Routing Protocols Library for more information about using the `auto-export` statement. See "Configuring Overlapping VPNs Using Automatic Route Export" on page 867 for more examples of export policy.

With this configuration on hub PE, spoke-to-spoke CE traffic goes through the hub CE and permits egress features (such as filtering) to be enabled on the hub PE.

In this configuration example, traffic forwarding is as follows between spoke CE2 and spoke CE3:

1. Spoke CE2 forwards traffic using the default route learned from spoke PE2 through BGP.

```
0.0.0.0/0          *[BGP/170] 02:24:15, localpref 100
                      AS path: 200 200 I
                    > to 10.49.3.1 via t1-3/0/1.0
```

2. Spoke PE2 does a route lookup in the spoke VRF table and forwards the traffic to hub PE1 (through the P router—PE2 pushes two labels) using the default route learned through BGP.

```
spoke.inet.0: 5 destinations, 5 routes (5 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

0.0.0.0/0          *[BGP/170] 00:00:09, localpref 100, from 10.255.14.176
                      AS path: 100 I
                    > via t3-0/0/0.0, Push 1029, Push 100224(top)
```

3. Hub PE1 does a route lookup in the mpls.0 table for the VPN label `1029`.

```
mpls.0: 5 destinations, 5 routes (5 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both
1029               *[VPN/0] 00:11:49
                      to table hub_downstream.inet.0, Pop
```

   The VPN label `1029` is advertised because:

   a. The `vrf-table-label` statement is applied at the `[edit routing-instances hub_downsteam]` hierarchy level in the hub PE1 configuration.

   b. The `no-vrf-advertise` statement is applied at the `[edit routing-instances hub]` hierarchy level, instructing the router to advertise the route from the secondary table.

   Therefore, IP lookups are performed in the hub_downstream.inet.0 table, not in the hub.inet.0 table.

Issue the `show route advertising-protocol` command on the hub PE to a spoke PE to verify the VPN label `1029` advertisement:

```
user@host> show route advertising-protocol
    hub_downstream.inet.0: 2 destinations, 2 routes (2 active, 0 holddown, 0 hidden)
    * 0.0.0.0/0 (1 entry, 1 announced)
     BGP group ibgp type Internal
         Route Distinguisher: 10.255.14.176:3
         VPN Label: 1029
         Nexthop: Self
         Localpref: 100
         AS path: 100 I
         Communities: target:200:101
```

4. Hub PE1 performs an IP lookup in the `hub_downstream.inet.0` table and forwards the traffic out interface `t3-0/0/0.0` to hub CE1.

```
    hub_downstream.inet.0: 2 destinations, 2 routes (2 active, 0 holddown, 0 hidden)
    0.0.0.0/0 (1 entry, 1 announced)
            *BGP    Preference: 170/-101
                    Next-hop reference count: 4
                    Source: 10.49.4.2
                    Next hop: 10.49.4.2 via t3-0/0/0.0, selected
                    State: <Secondary Active Ext>
                    Peer AS:   100
                    Age: 3:03
                    Task: BGP_100.10.49.4.2+1707
                    Announcement bits (2): 0-KRT 2-BGP.0.0.0.0+179
                    AS path: 100 I
                    Communities: target:200:101
                    Localpref: 100
                    Router ID: 10.49.10.251
                    Primary Routing Table hub.inet.0
```

The primary routing table is `hub.inet.0`, indicating that this route was exported from table `hub.inet.0` into this hub_downstream.inet.0 table as a result of the `no-vrf-advertise` statement at the `[edit routing-instances hub]` hierarchy level and the `auto-export` statement at the `[edit routing-instances hub-downstream routing-options]` hierarchy level in the hub PE1 configuration.

5. Hub CE1 forwards the traffic back to hub PE1 using the router learned through BGP.

```
10.49.10.253/32    *[BGP/170] 02:40:03, localpref 100
                      AS path: 200 200 I
                    > to 10.49.4.1 via t3-3/1/0.0
```

6. Hub PE1 performs a route lookup in the hub VRF table and forwards the traffic to spoke PE3 (through the P router—PE1 pushes two labels).

```
10.49.10.253/32    *[BGP/170] 01:41:05, localpref 100, from 10.255.14.178
                      AS path: 100 I
                    > via t1-0/1/0.0, Push 100128, Push 100192(top)
```

7. Spoke PE3 performs a route lookup in the mpls.0 table for VPN label `100128`.

```
100128             *[VPN/170] 02:41:30
                    > to 10.49.6.2 via t3-0/0/0.0, Pop
```

8. Spoke PE3 forwards traffic out interface `t3-0/0/0.0` to spoke CE3.

Configuring Hub-and-Spoke VPN Topologies: Two Interfaces

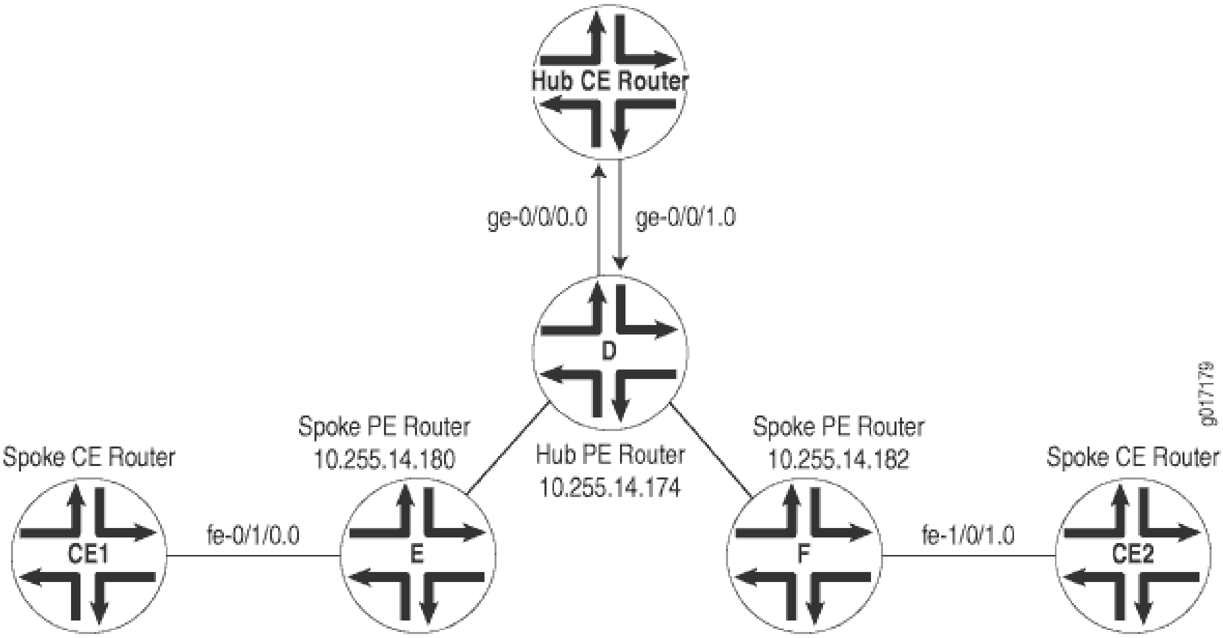
IN THIS SECTION

Use a two-interface configuration to propagate routes from spoke to spoke.

The example in this section configures a hub-and-spoke topology with two interfaces using the following components (see Figure 65 on page 833):

- One hub PE router (Router D).
- One hub CE router connected to the hub PE router. For this hub-and-spoke VPN topology to function properly, there must be two interfaces connecting the hub PE router to the hub CE router, and each interface must have its own VRF table on the PE router:
  - The first interface (here, interface ge-0/0/0.0) is used to announce spoke routes to the hub CE router. The VRF table associated with this interface contains the routes being announced by the spoke PE routers to the hub CE router.
  - The second interface (here, interface ge-0/0/1.0) is used to receive route announcements from the hub CE that are destined for the hub-and-spoke routers. The VRF table associated with this interface contains the routes announced by the hub CE router to the spoke PE routers. For this example, two separate physical interfaces are used. It would also work if you were to configure two separate logical interfaces sharing the same physical interface between the hub PE router and the hub CE router.
- Two spoke PE routers (Router E and Router F).
- Two spoke CE routers (CE1 and CE2), one connected to each spoke PE router.
- LDP as the signaling protocol.

Figure 65: Example of a Hub-and-Spoke VPN Topology with Two Interfaces



In this configuration, route distribution from spoke CE Router CE1 occurs as follows:

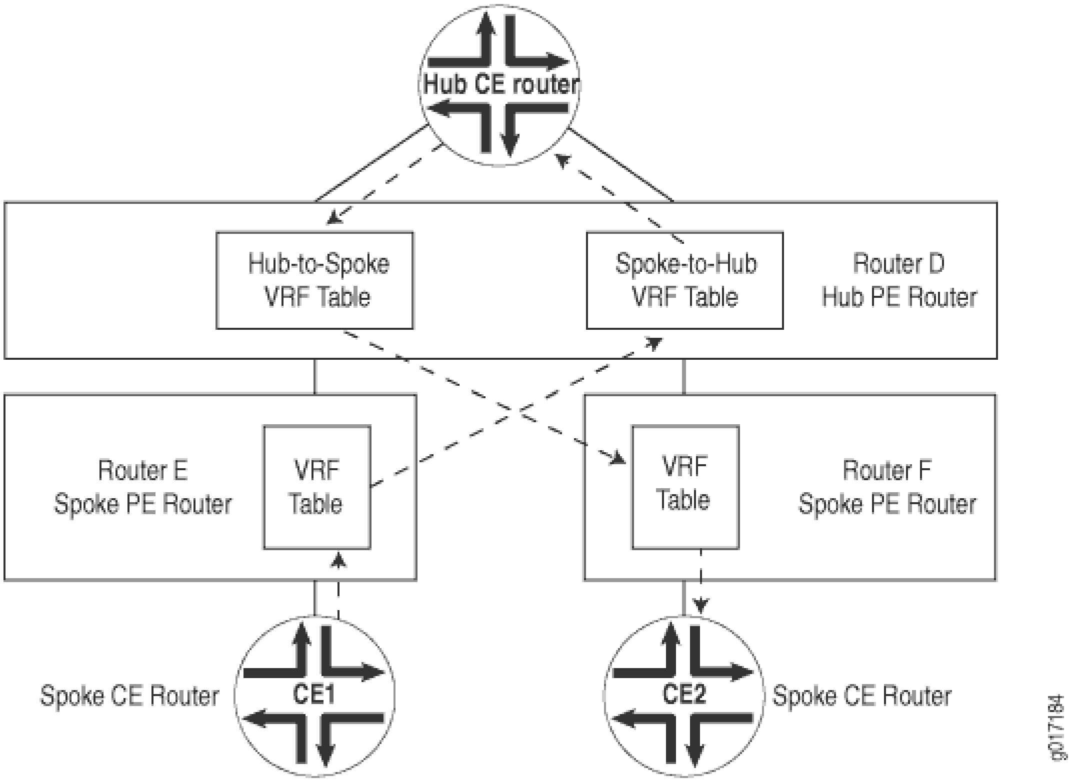
1. Spoke Router CE1 announces its routes to spoke PE Router E.

2. Router E installs the routes from CE1 into its VRF table.
3. After checking its VRF export policy, Router E adds the spoke target community to the routes from Router CE1 that passed the policy and announces them to the hub PE router, Router D.
4. Router D checks the VRF import policy associated with interface ge-0/0/0.0 and places all routes from spoke PE routers that match the policy into its bgp.l3vpn routing table. (Any routes that do not match are discarded.)
5. Router D checks its VRF import policy associated with interface ge-0/0/0.0 and installs all routes that match into its spoke VRF table. The routes are installed with the spoke target community.
6. Router D announces routes to the hub CE over interface ge-0/0/0.
7. The hub CE router announces the routes back to the hub PE Router D over the second interface to the hub router, interface ge-0/0/1.
8. The hub PE router installs the routes learned from the hub CE router into its hub VRF table, which is associated with interface ge-0/0/1.
9. The hub PE router checks the VRF export policy associated with interface ge-0/0/1.0 and announces all routes that match to all spokes after adding the hub target community.

Figure 66 on page 835 illustrates how routes are distributed from this spoke router to the other spoke CE router, Router CE2. The same path is followed if you issue a `traceroute` command from Router CE1 to Router CE2.

The final section in this example, "Hub-and-Spoke VPN Configuration Summarized by Router" on page 845, consolidates the statements needed to configure VPN functionality for each of the service provider routers shown in Figure 65 on page 833.

Figure 66: Route Distribution Between Two Spoke Routers



The following sections explain how to configure the VPN functionality for a hub-and-spoke topology on the hub-and-spoke PE routers. The CE routers do not have any information about the VPN, so you configure them normally.

Enabling an IGP on the Hub-and-Spoke PE Routers

To allow the hub-and-spoke PE routers to exchange routing information, you must configure an IGP on all these routers or you must configure static routes. You configure the IGP on the primary instance of the routing protocol process (rpd) (that is, at the `[edit protocols]` hierarchy level), not within the routing instance (that is, not at the `[edit routing-instances]` hierarchy level).

You configure the IGP in the standard way. This configuration example does not include this portion of the configuration.

In the route distribution in a hub-and-spoke topology, if the protocol used between the CE and PE routers at the hub site is BGP, the hub CE router announces all routes received from the hub PE router and the spoke routers back to the hub PE router and all the spoke routers. This means that the hub-and-spoke PE routers receive routes that contain their AS number. Normally, when a route contains this information, it indicates that a routing loop has occurred and the router rejects the routes. However, for the VPN configuration to work, the hub PE router and the spoke routers must accept these routes. To enable this, include the `loops` option when configuring the AS at the `[edit routing-options]` hierarchy level on the hub PE router and all the spoke routers. For this example configuration, you specify a value of 1. You can specify a number from 0 through 10.

```
[edit routing-options]
autonomous-system as-number loops 1;
```

Configuring LDP on the Hub-and-Spoke PE Routers

Configure LDP on the interfaces between the hub-and-spoke PE routers that participate in the VPN.

On hub PE Router D, configure LDP:

```
[edit protocols]
ldp {
    interface so-1/0/0.0;
    interface t3-1/1/0.0;
}
```

On spoke PE Router E, configure LDP:

```
[edit protocols]
ldp {
    interface fe-0/1/2.0;
}
```

On spoke PE router Router F, configure LDP:

```
[edit protocols]
ldp {
    interface fe-1/0/0.0;
}
```

Configuring IBGP on the PE Routers

On the hub-and-spoke PE routers, configure an IBGP session with the following properties:

- VPN family—To indicate that the IBGP session is for the VPN, include the `family inet-vpn` statement.
- Loopback address—Include the `local-address` statement, specifying the local PE router's loopback address. The IBGP session for VPNs runs through the loopback address. You must also configure the

`lo0` interface at the `[edit interfaces]` hierarchy level. The example does not include this part of the router's configuration.

- Neighbor address—Include the `neighbor` statement. On the hub router, specify the IP address of each spoke PE router, and on the spoke router, specify the address of the hub PE router.

For the hub router, you configure an IBGP session with each spoke, and for each spoke router, you configure an IBGP session with the hub. There are no IBGP sessions between the two spoke routers.

On hub Router D, configure IBGP. The first `neighbor` statement configures an IBGP session to spoke Router E, and the second configures a session to spoke Router F.

```
[edit protocols]
bgp {
    group Hub-to-Spokes {
        type internal;
        local-address 10.255.14.174;
        family inet-vpn {
            unicast;
        }
        neighbor 10.255.14.180;
        neighbor 10.255.14.182;
    }
}
```

On spoke Router E, configure an IBGP session to the hub router:

```
[edit protocols]
bgp {
    group Spoke-E-to-Hub {
        type internal;
        local-address 10.255.14.180;
        neighbor 10.255.14.174 {
            family inet-vpn {
                unicast;
            }
        }
    }
}
```

On spoke Router F, configure an IBGP session to the hub router:

```
[edit protocols]
bgp {
    group Spoke-F-to-Hub {
        type internal;
        local-address 10.255.14.182;
        neighbor 10.255.14.174 {
            family inet-vpn {
                unicast;
            }
        }
    }
}
```

Configuring VPN Routing Instances on the Hub-and-Spoke PE Routers

For the hub PE router to be able to distinguish between packets going to and coming from the spoke PE routers, you must configure it with two routing instances:

- One routing instance (in this example, `Spokes-to-Hub-CE`) is associated with the interface that carries packets from the hub PE router to the hub CE router (in this example, interface `ge-0/0/0.0`). Its VRF table contains the routes being announced by the spoke PE routers and the hub PE router to the hub CE router.

- The second routing instance (in this example, `Hub-CE-to-Spokes`) is associated with the interface that carries packets from the hub CE router to the hub PE router (in this example, interface `ge-0/0/1.0`). Its VRF table contains the routes being announced from the hub CE router to the hub-and-spoke PE routers.

On each spoke router, you must configure one routing instance.

You must define the following in the routing instance:

- Route distinguisher, which is used to distinguish the addresses in one VPN from those in another VPN.

- Instance type of `vrf`, which creates the VRF table on the PE router.

- Interfaces that are part of the VPN and that connect the PE routers to their CE routers.

- VRF import and export policies. Both import policies must include reference to a community. Otherwise, when you try to commit the configuration, the commit fails. (The exception to this is if the import policy contains only a `then reject` statement.) In the VRF export policy, spoke PE routers attach the spoke target community.

- Routing between the PE and CE routers, which is required for the PE router to distribute VPN-related routes to and from connected CE routers. You can configure a routing protocol—BGP, OSPF, or RIP—or you can configure static routing.

For a hub-and-spoke topology, you must configure different policies in each routing instance on the hub CE router. For the routing instance associated with the interface that carries packets from the hub PE router to the hub CE router (in this example, `Spokes-to-Hub-CE`), the import policy must accept all routes received on the IBGP session between the hub-and-spoke PE routers, and the export policy must reject all routes received from the hub CE router. For the routing instance associated with the interface that carries packets from the hub CE router to the hub PE router (in this example, `Hub-CE-to-Spokes`), the import policy must reject all routes received from the spoke PE routers, and the export policy must export to all the spoke routers.

On hub PE Router D, configure the following routing instances. Router D uses OSPF to distribute routes to and from the hub CE router.

```
[edit]
routing-instance {
    Spokes-to-Hub-CE {
        instance-type vrf;
        interface ge-0/0/0.0;
        route-distinguisher 10.255.1.174:65535;
        vrf-import spoke;
        vrf-export null;
        protocols {
            ospf {
                domain-id disable;
                export redistribute-vpn;
                domain-vpn-tag 0;
                area 0.0.0.0 {
                    interface ge-0/0/0;
                }
            }
        }
    }
    Hub-CE-to-Spokes {
        instance-type vrf;
        interface ge-0/0/1.0;
        route-distinguisher 10.255.1.174:65534;
        vrf-import null;
```

```
        vrf-export hub;
        protocols {
            ospf {
                export redistribute-vpn;
                area 0.0.0.0 {
                    interface ge-0/0/1.0;
                }
            }
        }
    }
}
```

On spoke PE Router E, configure the following routing instances. Router E uses OSPF to distribute routes to and from spoke CE Router CE1.

```
[edit]
routing-instance {
    Spoke-E-to-Hub {
        instance-type vrf;
        interface fe-0/1/0.0;
        route-distinguisher 10.255.14.80:65035;
        vrf-import hub;
        vrf-export spoke;
        protocols {
            ospf {
                export redistribute-vpn;
                area 0.0.0.0 {
                    interface fe-0/1/0.0;
                }
            }
        }
    }
}
```

On spoke PE Router F, configure the following routing instances. Router F uses OSPF to distribute routes to and from spoke CE Router CE2.

```
[edit]
routing-instance {
    Spoke-F-to-Hub {
        instance-type vrf;
```

```
        interface fe-1/0/1.0;
        route-distinguisher 10.255.14.182:65135;
        vrf-import hub;
        vrf-export spoke;
        protocols {
            ospf {
                export redistribute-vpn;
                area 0.0.0.0 {
                    interface fe-1/0/1.0;
                }
            }
        }
    }
}
```

Configuring VPN Policy on the PE Routers

You must configure VPN import and export policies on each of the hub-and-spoke PE routers so that they install the appropriate routes in the VRF tables, which they use to forward packets within each VPN.

On the spoke routers, you define policies to exchange routes with the hub router.

On the hub router, you define policies to accept routes from the spoke PE routers and distribute them to the hub CE router, and vice versa. The hub PE router has two VRF tables:

- Spoke-to-hub VRF table—Handles routes received from spoke routers and announces these routes to the hub CE router. For this VRF table, the import policy must check that the spoke target name is present and that the route was received from the IBGP session between the hub PE and the spoke PE routers. This VRF table must not export any routes, so its export policy should reject everything.

- Hub-to-spoke VRF table—Handles routes received from the hub CE router and announces them to the spoke routers. For this VRF table, the export policy must add the hub target community. This VRF table must not import any routes, so its import policy should reject everything.

In the VPN policy, you also configure the VPN target communities.

On hub PE Router D, configure the following policies to apply to the VRF tables:

- `spoke`—Accepts routes received from the IBGP session between it and the spoke PE routers that contain the community target `spoke`, and rejects all other routes.

- `hub`—Adds the community target hub to all routes received from OSPF (that is, from the session between it and the hub CE router). It rejects all other routes.

- `null`—Rejects all routes.
- `redistribute-vpn`—Redistributes OSPF routes to neighbors within the routing instance.

```
[edit]
policy-options {
    policy-statement spoke {
        term a {
            from {
                protocol bgp;
                community spoke;
            }
            then accept;
        }
        term b {
            then reject;
        }
    }
    policy-statement hub {
        term a {
            from protocol ospf;
            then {
                community add hub;
                accept;
            }
        }
        term b {
            then reject;
        }
    }
    policy-statement null {
        then reject;
    }
    policy-statement redistribute-vpn {
        term a {
            from protocol bgp;
            then accept;
        }
        term b {
            then reject;
        }
    }
```

```
    community hub members target:65535:1;
    community spoke members target:65535:2;
}
```

To apply the VRF policies on Router D, include the `vrf-export` and `vrf-import` statements when you configure the routing instances:

```
[edit]
routing-instance {
    Spokes-to-Hub-CE {
        vrf-import spoke;
        vrf-export null;
    }
    Hub-CE-to-Spokes {
        vrf-import null;
        vrf-export hub;
    }
}
```

On spoke PE Router E and Router F, configure the following policies to apply to the VRF tables:

- `hub`—Accepts routes received from the IBGP session between it and the hub PE routers that contain the community target `hub`, and rejects all other routes.
- `spoke`—Adds the community target spoke to all routes received from OSPF (that is, from the session between it and the hub CE router) rejects all other routes.
- `redistribute-vpn`—Redistributes OSPF routes to neighbors within the routing instance.

On spoke PE Router E and Router F, configure the following VPN import and export policies:

```
[edit]
policy-options {
    policy-statement hub {
        term a {
            from {
                protocol bgp;
                community hub;
            }
            then accept;
        }
        term b {
```

```
            then reject;
        }
    }
    policy-statement spoke {
        term a {
            from protocol ospf;
            then {
                community add spoke;
                accept;
            }
        }
        term b {
            then reject;
        }
    }
    policy-statement redistribute-vpn {
        term a {
            from protocol bgp;
            then accept;
        }
        term b {
            then reject;
        }
    }
    community hub members target:65535:1;
    community spoke members target 65535:2;
}
```

To apply the VRF policies on the spoke routers, include the `vrf-export` and `vrf-import` statements when you configure the routing instances:

```
[edit]
routing-instance {
    Spoke-E-to-Hub {
        vrf-import hub;
        vrf-export spoke;
    }
}
[edit]
routing-instance {
    Spoke-F-to-Hub {
        vrf-import hub;
```

```
        vrf-export spoke;
    }
}
```

Hub-and-Spoke VPN Configuration Summarized by Router

Router D (Hub PE Router)

Routing Instance for Distributing Spoke Routes to Hub CE

```
Spokes-to-Hub-CE {
    instance-type vrf;
    interface fe-0/0/0.0;
    route-distinguisher 10.255.1.174:65535;
    vrf-import spoke;
    vrf-export null;
}
```

Instance Routing Protocol

```
protocols {
    ospf {
        domain-id disable;
        domain-vpn-tag 0;
        export redistribute-vpn;
        area 0.0.0.0 {
            interface ge-0/0/0.0;
        }
    }
}
```

Routing Instance for Distributing Hub CE Routes to Spokes

```
Hub-CE-to-Spokes {
    instance-type vrf;
    interface ge-0/0/1.0;
    route-distinguisher 10.255.1.174:65534;
    vrf-import null;
```

```
    vrf-export hub;
}
```

Routing Instance Routing Protocols

```
protocols {
    ospf {
        export redistribute-vpn;
        area 0.0.0.0 {
            interface ge-0/0/1.0;
        }
    }
}
```

Routing Options (Primary Instance)

```
routing-options {
    autonomous-system 1 loops 1;
}
```

Protocols (Primary Instance)

```
protocols {
}
```

Enable LDP

```
ldp {
    interface so-1/0/0.0;
    interface t3-1/1/0.0;
}
```

Configure IBGP

```
bgp {
    group Hub-to-Spokes {
        type internal;
        local-address 10.255.14.174;
        family inet-vpn {
```

```
                unicast;
            }
            neighbor 10.255.14.180;
            neighbor 10.255.14.182;
        }
    }
```

Configure VPN Policy

```
policy-options {
    policy-statement spoke {
        term a {
            from {
                protocol bgp;
                community spoke;
            }
            then accept;
        }
        term b {
            then reject;
        }
    }
    policy-statement hub {
        term a {
            from protocol ospf;
            then {
                community add hub;
                accept;
            }
        }
        term b {
            then reject;
        }
    }
    policy-statement null {
        then reject;
    }
    policy-statement redistribute-vpn {
        term a {
            from protocol bgp;
            then accept;
        }
```

```
        term b {
            then reject;
        }
    }
    community hub members target:65535:1;
    community spoke members target:65535:2;
}
```

Router E (Spoke PE Router)

Routing Instance

```
routing-instance {
    Spoke-E-to-Hub {
        instance-type vrf;
        interface fe-0/1/0.0;
        route-distinguisher 10.255.14.80:65035;
        vrf-import hub;
        vrf-export spoke;
    }
}
```

Instance Routing Protocol

```
protocols {
    ospf {
        export redistribute-vpn;
        area 0.0.0.0 {
            interface fe-0/1/0.0;
        }
    }
}
```

Routing Options (Primary Instance)

```
routing-options {
    autonomous-system 1 loops 1;
}
```

Protocols (Primary Instance)

```
protocols {
}
```

Enable LDP

```
ldp {
    interface fe-0/1/2.0;
}
```

Configure IBGP

```
bgp {
    group Spoke-E-to-Hub {
        type internal;
        local-address 10.255.14.180;
        neighbor 10.255.14.174 {
            family inet-vpn {
                unicast;
            }
        }
    }
}
```

Configure VPN Policy

```
policy-options {
    policy-statement hub {
        term a {
            from {
                protocol bgp;
                community hub;
            }
            then accept;
        }
        term b {
            then reject;
        }
    }
```

```
    policy-statement spoke {
        term a {
            from protocol ospf;
            then {
                community add spoke;
                accept;
            }
        }
        term b {
            then reject;
        }
    }
    policy-statement redistribute-vpn {
        term a {
            from protocol bgp;
            then accept;
        }
        term b {
            then reject;
        }
    }
    community hub members target:65535:1;
    community spoke members target:65535:2;
}
```

Router F (Spoke PE Router)

Routing Instance

```
routing-instance {
    Spoke-F-to-Hub {
        instance-type vrf;
        interface fe-1/0/1.0;
        route-distinguisher 10.255.14.182:65135;
        vrf-import hub;
        vrf-export spoke;
    }
}
```

Instance Routing Protocol

```
protocols {
    ospf {
        export redistribute-vpn;
        area 0.0.0.0 {
            interface fe-1/0/1.0;
        }
    }
}
```

Routing Options (Primary Instance)

```
routing-options {
    autonomous-system 1 loops 1;
}
```

Protocols (Primary Instance)

```
protocols {
}
```

Enable LDP

```
ldp {
    interface fe-1/0/0.0;
}
```

Configure IBGP

```
bgp {
    group Spoke-F-to-Hub {
        type internal;
        local-address 10.255.14.182;
        neighbor 10.255.14.174 {
            family inet-vpn {
                unicast;
            }
        }
```

```
    }
}
```

Configure VPN Policy

```
policy-options {
    policy-statement hub {
        term a {
            from {
                protocol bgp;
                community hub;
            }
            then accept;
        }
        term b {
            then reject;
        }
    }
    policy-statement spoke {
        term a {
            from protocol ospf;
            then {
                community add spoke;
                accept;
            }
        }
        term b {
            then reject;
        }
    }
    policy-statement redistribute-vpn {
        term a {
            from {
                protocol bgp;
            }
            then accept;
        }
        term b {
            then reject;
        }
    }
    community hub members target:65535:1;
```

```
    community spoke members target:65535:2;
}
```

Overlapping VPNs

IN THIS SECTION

Configuring Overlapping VPNs Using Routing Table Groups

IN THIS SECTION

In Layer 3 VPNs, a CE router is often a member of more than one VPN. This example illustrates how to configure PE routers that support CE routers that support multiple VPNs. Support for this type of configuration uses a Junos OS feature called routing table groups (sometimes also called routing information base [RIB] groups), which allows a route to be installed into several routing tables. A routing table group is a list of routing tables into which the protocol should install its routes.

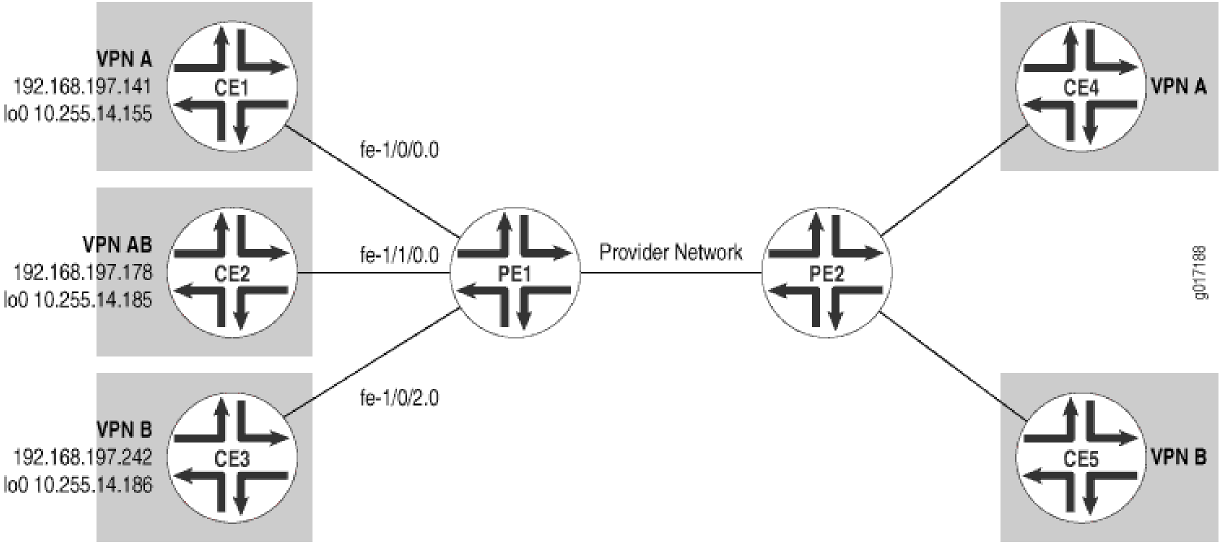
You define routing table groups at the `[edit routing-options]` hierarchy level for the default instance. You cannot configure routing table groups at the `[edit routing-instances routing-options]` hierarchy level; doing so results in a commit error.

After you define a routing table group, it can be used by multiple protocols. You can also apply routing table groups to static routing. The configuration examples in this section include both types of configurations.

Figure 67 on page 854 illustrates the topology for the configuration example in this section. The configurations in this section illustrate local connectivity between CE routers connected to the same PE router. If Router PE1 were connected only to Router CE2 (VPN AB), there would be no need for any extra configuration. The configuration statements in the sections that follow enable VPN AB Router CE2 to communicate with VPN A Router CE1 and VPN B Router CE3, which are directly connected to Router PE1. VPN routes that originate from the remote PE routers (the PE2 router in this case) are placed in a global Layer 3 VPN routing table (bgp.l3vpn.inet.0), and routes with appropriate route targets are imported into the routing tables as dictated by the VRF import policy configuration. The goal is to be able to choose routes from individual VPN routing tables that are locally populated.

Router PE1 is where all the filtering and configuration modification takes place. Therefore only VPN configurations for PE1 are shown. The CE routers do not have any information about the VPN, so you can configure them normally.

Figure 67: Example of an Overlapping VPN Topology



The following sections explain several ways to configure overlapping VPNs.

The following sections illustrate different scenarios for configuring overlapping VPNs, depending on the routing protocol used between the PE and CE routers. For all of these examples, you need to configure routing table groups.

Configuring Routing Table Groups

In this example, routing table groups are common in the four configuration scenarios. The routing table groups are used to install routes (including interface, static, OSPF, and BGP routes) into several routing tables for the default and other instances. In the routing table group definition, the first routing table is called the primary routing table. (Normally, the primary routing table is the table into which the route would be installed if you did not configure routing table groups. The other routing tables are called secondary routing tables.)

The routing table groups in this configuration install routes as follows:

- `vpna-vpnab` installs routes into routing tables VPN-A.inet.0 and VPN-AB.inet.0.
- `vpnb-vpnab` installs routes into routing tables VPN-B.inet.0 and VPN-AB.inet.0.
- `vpnab-vpna_and_vpnb` installs routes into routing tables VPN-AB.inet.0, VPN-A.inet.0, and VPN-B.inet.0.

Configure the routing table groups:

```
[edit]
routing-options {
    rib-groups {
        vpna-vpnab {
            import-rib [ VPN-A.inet.0 VPN-AB.inet.0 ];
        }
        vpnb-vpnab {
            import-rib [ VPN-B.inet.0 VPN-AB.inet.0 ];
        }
        vpnab-vpna_and_vpnb {
            import-rib [ VPN-AB.inet.0 VPN-A.inet.0 VPN-B.inet.0 ];
        }
    }
}
```

Configuring Static Routes Between the PE and CE Routers

To configure static routing between the PE1 router and the CE1, CE2, and CE3 routers, you must configure routing instances for VPN A, VPN B, and VPN AB (you configure static routing under each instance):

Configuring the Routing Instance for VPN A

On Router PE1, configure VPN A:

```
[edit]
routing-instances {
    VPN-A {
        instance-type vrf;
```

```
        interface fe-1/0/0.0;
        route-distinguisher 10.255.14.175:3;
        vrf-import vpna-import;
        vrf-export vpna-export;
        routing-options {
            interface-routes {
                rib-group inet vpna-vpnab;
            }
            static {
                route 10.255.14.155/32 next-hop 192.168.197.141;
                route 10.255.14.185/32 next-hop 192.168.197.178;
            }
        }
    }
}
```

The `interface-routes` statement installs VPN A’s interface routes into the routing tables defined in the routing table group `vpna-vpnab`.

The `static` statement configures the static routes that are installed in the VPN-A.inet.0 routing table. The first static route is for Router CE1 (VPN A) and the second is for Router CE2 (in VPN AB).

Next hop `192.168.197.178` is not in VPN A. Route `10.255.14.185/32` cannot be installed in VPN-A.inet.0 unless interface routes from routing instance VPN AB are installed in this routing table. Including the `interface-routes` statements in the VPN AB configuration provides this next hop. Similarly, including the `interface-routes` statement in the VPN AB configuration installs `192.168.197.141` into VPN-AB.inet.0.

Configuring the Routing Instance for VPN AB

On Router PE1, configure VPN AB:

```
[edit]
routing instances {
    VPN-AB {
        instance-type vrf;
        interface fe-1/1/0.0;
        route-distinguisher 10.255.14.175:9;
        vrf-import vpnab-import;
        vrf-export vpnab-export;
        routing-options {
            interface-routes {
                rib-group vpnab-vpna_and_vpnb;
```

```
            }
            static {
                route 10.255.14.185/32 next-hop 192.168.197.178;
                route 10.255.14.155/32 next-hop 192.168.197.141;
                route 10.255.14.186/32 next-hop 192.168.197.242;
            }
        }
    }
}
```

In this configuration, the following static routes are installed in the VPN-AB.inet.0 routing table:

- `10.255.14.185/32` is for Router CE2 (in VPN AB)
- `10.255.14.155/32` is for Router CE1 (in VPN A)
- `10.255.14.186/32` is for Router CE3 (in VPN B)

Next hops `192.168.197.141` and `192.168.197.242` do not belong to VPN AB. Routes `10.255.14.155/32` and `10.255.14.186/32` cannot be installed in VPN-AB.inet.0 unless interface routes from VPN A and VPN B are installed in this routing table. The interface route configurations in VPN A and VPN B routing instances provide these next hops.

Configuring the Routing Instance for VPN B

On Router PE1, configure VPN B:

```
[edit]
routing instances {
    VPN-B {
        instance-type vrf;
        interface fe-1/0/2.0;
        route-distinguisher 10.255.14.175:10;
        vrf-import vpnb-import;
        vrf-export vpnb-export;
        routing-options {
            interface-routes {
                rib-group inet vpnb-vpnab;
            }
            static {
                route 10.255.14.186/32 next-hop 192.168.197.242;
                route 10.255.14.185/32 next-hop 192.168.197.178;
            }
```

```
        }
    }
}
```

When you configure the routing instance for VPN B, these static routes are placed in VPNB.inet.0:

- `10.255.14.186/32` is for Router CE3 (in VPN B)
- `10.255.14.185/32` is for Router CE2 (in VPN AB)

Next hop `192.168.197.178` does not belong to VPN B. Route `10.255.14.185/32` cannot be installed in VPN-B.inet.0 unless interface routes from VPN AB are installed in this routing table. The interface route configuration in VPN AB provides this next hop.

Configuring VPN Policy

The `vrf-import` and `vrf-export` policy statements that you configure for overlapping VPNs are the same as policy statements for regular VPNs, except that you include the `from interface` statement in each VRF export policy. This statement forces each VPN to announce only those routes that originated from that VPN. For example, VPN A has routes that originated in VPN A and VPN AB. If you do not include the `from interface` statement, VPN A announces its own routes as well as VPN AB's routes, so the remote PE router receives multiple announcements for the same routes. Including the `from interface` statement restricts each VPN to announcing only the routes it originated and allows you to filter out the routes imported from other routing tables for local connectivity.

In this configuration example, the `vpnab-import` policy accepts routes from VPN A, VPN B, and VPN AB. The `vpna-export` policy exports only routes that originate in VPN A. Similarly, the `vpnb-export` and `vpnab-export` policies export only routes that originate within the respective VPNs.

On Router PE1, configure the following VPN import and export policies:

```
[edit]
policy-options {
    policy-statement vpna-import {
        term a {
            from {
                protocol bgp;
                community VPNA-comm;
            }
            then accept;
        }
        term b {
            then reject;
```

```
        }
    }
    policy-statement vpnb-import {
        term a {
            from {
                protocol bgp;
                community VPNB-comm;
            }
            then accept;
        }
        term b {
            then reject;
        }
    }
    policy-statement vpnab-import {
        term a {
            from {
                protocol bgp;
                community [ VPNA-comm VPNB-comm ];
            }
            then accept;
        }
        term b {
            then reject;
        }
    }
    policy-statement vpna-export {
        term a {
            from {
                protocol static;
                interface fe-1/0/0.0;
            }
            then {
                community add VPNA-comm;
                accept;
            }
        }
        term b {
            then reject;
        }
    }
    policy-statement vpnb-export {
        term a {
```

```
            from {
                protocol static;
                interface fe-1/0/2.0;
            }
            then {
                community add VPNB-comm;
                accept;
            }
        }
        term b {
            then reject;
        }
    }
    policy-statement vpnab-export {
        term a {
            from {
                protocol static;
                interface fe-1/1/0.0;
            }
            then {
                community add VPNB-comm;
                community add VPNA-comm;
                accept;
            }
        }
        term b {
            then reject;
        }
    }
    community VPNA-comm members target:69:1;
    community VPNB-comm members target:69:2;
}
```

On Router PE1, apply the VPN import and export policies:

```
[edit]
routing-instances {
    VPN-A {
        instance-type vrf;
        interface fe-1/0/0.0;
        route-distinguisher 10.255.14.175:3;
        vrf-import vpna-import;
```

```
        vrf-export vpna-export;
        routing-options {
            static {
                rib-group vpna-vpnab;
                route 10.255.14.155/32 next-hop 192.168.197.141;
                route 10.255.14.185/32 next-hop 192.168.197.178;
            }
        }
    }
    VPN-AB {
        instance-type vrf;
        interface fe-1/1/0.0;
        route-distinguisher 10.255.14.175:9;
        vrf-import vpnab-import;
        vrf-export vpnab-export;
        routing-options {
            static {
                rib-group vpnab-vpna_and_vpnb;
                route 10.255.14.185/32 next-hop 192.168.197.178;
            }
        }
    }
    VPN-B {
        instance-type vrf;
        interface fe-1/0/2.0;
        route-distinguisher 10.255.14.175:10;
        vrf-import vpnb-import;
        vrf-export vpnb-export;
        routing-options {
            static {
                rib-group vpnb-vpnab;
                route 10.255.14.186/32 next-hop 192.168.197.242;
            }
        }
    }
}
```

For VPN A, include the `routing-options` statement at the `[edit routing-instances` *`routing-instance-name`*`]` hierarchy level to install the static routes directly into the routing tables defined in the routing table group `vpna-vpnab`. For VPN AB, the configuration installs the static route directly into the routing tables defined in the routing table group `vpnab-vpna` and `vpnab-vpnb`. For VPN B the configuration installs the static route directly into the routing tables defined in the routing table group `vpnb-vpnab`.

Configuring BGP Between the PE and CE Routers

In this configuration example, the `vpna-site1` BGP group for VPN A installs the routes learned from the BGP session into the routing tables defined in the `vpna-vpnab` routing table group. For VPN AB, the `vpnab-site1` group installs the routes learned from the BGP session into the routing tables defined in the `vpnab-vpna_and_vpnb` routing table group. For VPN B, the `vpnb-site1` group installs the routes learned from the BGP session into the routing tables defined in the `vpnb-vpnab` routing table group. Interface routes are not needed for this configuration.

The VRF import and export policies are similar to those defined in "Configuring Static Routes Between the PE and CE Routers" on page 855, except the export protocol is BGP instead of a static route. On all `vrf-export` policies, you use the `from protocol bgp` statement.

On Router PE1, configure BGP between the PE and CE routers:

```
[edit]
routing-instances {
    VPN-A {
        instance-type vrf;
        interface fe-1/0/0.0;
        route-distinguisher 10.255.14.175:3;
        vrf-import vpna-import;
        vrf-export vpna-export;
        protocols {
            bgp {
                group vpna-site1 {
                    family inet {
                        unicast {
                            rib-group vpna-vpnab;
                        }
                    }
                    peer-as 1;
                    neighbor 192.168.197.141;
                }
            }
        }
    }
    VPN-AB {
        instance-type vrf;
        interface fe-1/1/0.0;
        route-distinguisher 10.255.14.175:9;
        vrf-import vpnab-import;
        vrf-export vpnab-export;
```

```
        protocols {
            bgp {
                group vpnab-site1 {
                    family inet {
                        unicast {
                            rib-group vpnab-vpna_and_vpnb;
                        }
                    }
                    peer-as 9;
                    neighbor 192.168.197.178;
                }
            }
        }
    }
    VPN-B {
        instance-type vrf;
        interface fe-1/0/2.0;
        route-distinguisher 10.255.14.175:10;
        vrf-import vpnb-import;
        vrf-export vpnb-export;
        protocols {
            bgp {
                group vpnb-site1 {
                    family inet {
                        unicast {
                            rib-group vpnb-vpnab;
                        }
                    }
                    neighbor 192.168.197.242 {
                        peer-as 10;
                    }
                }
            }
        }
    }
}
```

Configuring OSPF Between the PE and CE Routers

In this configuration example, routes learned from the OSPF session for VPN A are installed into the routing tables defined in the `vpna-vpnab` routing table group. For VPN AB, routes learned from the OSPF session are installed into the routing tables defined in the `vpnab-vpna_and_vpnb` routing table group. For

VPN B, routes learned from the OSPF session are installed into the routing tables defined in the `vpnb-vpnab` routing table group.

The VRF import and export policies are similar to those defined in "Configuring Static Routes Between the PE and CE Routers" on page 855 and "Configuring BGP Between the PE and CE Routers" on page 862, except the export protocol is OSPF instead of BGP or a static route. Therefore, on all `vrf-export` policies, you use the `from protocol ospf` statement instead of the `from protocol <static | bgp>` statement.

On Router PE1, configure OSPF between the PE and CE routers:

```
[edit]
routing-instances {
    VPN-A {
        instance-type vrf;
        interface fe-1/0/0.0;
        route-distinguisher 10.255.14.175:3;
        vrf-import vpna-import;
        vrf-export vpna-export;
        protocols {
            ospf {
                rib-group vpna-vpnab;
                export vpna-import;
                area 0.0.0.0 {
                    interface fe-1/0/0.0;
                }
            }
        }
    }
    VPN-AB {
        instance-type vrf;
        interface fe-1/1/0.0;
        route-distinguisher 10.255.14.175:9;
        vrf-import vpnab-import;
        vrf-export vpnab-export;
        protocols {
            ospf {
                rib-group vpnab-vpna_and_vpnb;
                export vpnab-import;
                area 0.0.0.0 {
                    interface fe-1/1/0.0;
                }
            }
        }
```

```
    }
    VPN-B {
        instance-type vrf;
        interface fe-1/0/2.0;
        route-distinguisher 10.255.14.175:10;
        vrf-import vpnb-import;
        vrf-export vpnb-export;
        protocols {
            ospf {
                rib-group vpnb-vpnab;
                export vpnb-import;
                area 0.0.0.0 {
                    interface fe-1/0/2.0;
                }
            }
        }
    }
}
```

Configuring Static, BGP, and OSPF Routes Between PE and CE Routers

This section shows how to configure the routes between the PE and CE routers by using a combination of static routes, BGP, and OSPF:

- The connection between Router PE1 and Router CE1 uses static routing.
- The connection between Router PE1 and Router CE2 uses BGP.
- The connection between Router PE1 and Router CE3 uses OSPF.

Here, the configuration for VPN AB also includes a static route to CE1.

On Router PE1, configure a combination of static routing, BGP, and OSPF between the PE and CE routers:

```
[edit]
routing-instances {
    VPN-A {
        instance-type vrf;
        interface fe-1/0/0.0;
        route-distinguisher 10.255.14.175:3;
        vrf-import vpna-import;
        vrf-export vpna-export;
```

```
        routing-options {
            static {
                rib-group vpna-vpnab;
                route 10.255.14.155/32 next-hop 192.168.197.141;
            }
        }
    }
    VPN-AB {
        instance-type vrf;
        interface fe-1/1/0.0;
        route-distinguisher 10.255.14.175:9;
        vrf-import vpnab-import;
        vrf-export vpnab-export;
        protocols {
            bgp {
                group vpnab-site1 {
                    family inet {
                        unicast {
                            rib-group vpnab-vpna_and_vpnb;
                        }
                    }
                    export to-vpnab-site1;
                    peer-as 9;
                    neighbor 192.168.197.178;
                }
            }
        }
    }
    VPN-B {
        instance-type vrf;
        interface fe-1/0/2.0;
        route-distinguisher 10.255.14.175:10;
        vrf-import vpnb-import;
        vrf-export vpnb-export;
        protocols {
            ospf {
                rib-group vpnb-vpnab;
                export vpnb-import;
                area 0.0.0.1 {
                    interface t3-0/3/3.0;
                }
            }
        }
```

```
    }
}
policy-options {
    policy-statement to-vpnab-site1 {
        term a {
            from protocol static;
            then accept;
        }
        term b {
            from protocol bgp;
            then accept;
        }
        term c {
            then reject;
        }
    }
}
```

Configuring Overlapping VPNs Using Automatic Route Export

IN THIS SECTION

A problem with multiple routing instances is how to export routes between routing instances. You can accomplish this in Junos OS by configuring routing table groups for each routing instance that needs to export routes to other routing tables. For information about how to configure overlapping VPNs by using routing table groups, see "Configuring Overlapping VPNs Using Routing Table Groups" on page 853.

However, using routing table groups has limitations:

- Routing table group configuration is complex. You must define a unique routing table group for each routing instance that will export routes.

- You must also configure a unique routing table group for each protocol that will export routes.

To limit and sometimes eliminate the need to configure routing table groups in multiple routing instance topologies, you can use the functionality provided by the `auto-export` statement.

The `auto-export` statement is particularly useful for configuring overlapping VPNs—VPN configurations where more than one VRF routing instance lists the same community route target in its `vrf-import` policy. The `auto-export` statement finds out which routing tables to export routes from and import routes to by examining the existing policy configuration.

The `auto-export` statement automatically exports routes between the routing instances referencing a given route target community. When the `auto-export` statement is configured, a VRF target tree is constructed based on the `vrf-import` and `vrf-export` policies configured on the system. If a routing instance references a route target in its `vrf-import` policy, the route target is added to the import list for the target. If it references a specific route target in its `vrf-export` policy, the route target is added to the export list for that target. Route targets where there is a single importer that matches a single exporter or with no importers or exporters are ignored.

Changes to routing tables that export route targets are tracked. When a route change occurs, the routing instance's `vpn-export` policy is applied to the route. If it is allowed, the route is imported to all the import tables (subject to the `vrf-import` policy) of the route targets set by the export policy.

The sections that follow describe how to configure overlapping VPNs by using the `auto-export` statement for inter-instance export in addition to routing table groups:

Configuring Overlapping VPNs with BGP and Automatic Route Export

The following example provides the configuration for an overlapping VPN where BGP is used between the PE and CE routers.

Configure routing instance `VPN-A`:

```
[edit]
routing-instances {
    VPN-A {
        instance-type vrf;
        interface fe-1/0/0.0;
        route-distinguisher 10.255.14.175:3;
        vrf-import vpna-import;
        vrf-export vpna-export;
        routing-options {
            auto-export;
        }
        protocols {
            bgp {
                group vpna-site1 {
```

```
                    peer-as 1;
                    neighbor 192.168.197.141;
                }
            }
        }
    }
}
```

Configure routing instance `VPN-AB`:

```
[edit]
routing-instances {
    VPN-AB {
        instance-type vrf;
        interface fe-1/1/0.0;
        route-distinguisher 10.255.14.175:9;
        vrf-import vpnab-import;
        vrf-export vpnab-export;
        routing-options {
            auto-export;
        }
        protocols {
            bgp {
                group vpnab-site1 {
                    peer-as 9;
                    neighbor 192.168.197.178;
                }
            }
        }
    }
}
```

For this configuration, the `auto-export` statement replaces the functionality that was provided by a routing table group configuration. However, sometimes additional configuration is required.

Since the `vrf-import` policy and the `vrf-export` policy from which the `auto-export` statement deduces the import and export matrix are configured on a per-instance basis, you must be able to enable or disable them for unicast and multicast, in case multicast network layer reachability information (NLRI) is configured.

Configuring Overlapping VPNs and Additional Tables

You might need to use the `auto-export` statement between overlapping VPNs but require that a subset of the routes learned from a VRF table be installed into the inet.0 table or in routing-instance.inet.2.

To support this type of scenario, where not all of the information needed is present in the `vrf-import` and `vrf-export` policies, you configure an additional list of routing tables by using an additional routing table group.

To add routes from `VPN-A` and `VPN-AB` to inet.0 in the example described, you need to include the following additional configuration statements:

Configure the routing options:

```
[edit]
routing-options {
    rib-groups {
        inet-access {
            import-rib inet.0;
        }
    }
}
```

Configure routing instance `VPN-A`:

```
[edit]
routing-instances {
    VPN-A {
        routing-options {
            auto-export {
                family inet {
                    unicast {
                        rib-group inet-access;
                    }
                }
            }
        }
    }
}
```

Configure routing instance `VPN-AB`:

```
[edit]
routing-instances {
    VPN-AB {
        routing-options {
            auto-export {
                family inet {
                    unicast {
                        rib-group inet-access;
                    }
                }
            }
        }
    }
}
```

Routing table groups are used in this configuration differently from how they are generally used in Junos OS. Routing table groups normally require that the exporting routing table be referenced as the primary import routing table in the routing table group. For this configuration, the restriction does not apply. The routing table group functions as an additional list of tables to which to export routes.

Configuring Automatic Route Export for All VRF Instances

The following configuration allows you to configure the `auto-export` statement for all of the routing instances in a configuration group:

```
[edit]
groups {
    vrf-export-on {
        routing-instances {
            <*> {
                routing-options {
                    auto-export;
                }
            }
        }
    }
}
apply-groups vrf-export-on;
```

CHAPTER 6

Layer 3 VPN Tunnels

IN THIS CHAPTER

ES Tunnels for Layer 3 VPNs

IN THIS SECTION

Configuring an ES Tunnel Interface for Layer 3 VPNs

IN THIS SECTION

An ES tunnel interface allows you to configure an IP Security (IPsec) tunnel between the PE and CE routers of a Layer 3 VPN. The IPsec tunnel can include one or more hops.

The following sections explain how to configure an ES tunnel interface between the PE and CE routers of a Layer 3 VPN:

Configuring the ES Tunnel Interface on the PE Router

To configure the ES tunnel interface on the PE router, include the `unit` statement:

```
unit logical-unit-number {
    tunnel {
        source source-address;
        destination destination-address;
    }
    family inet {
        address address;
        ipsec-sa security-association-name;
    }
}
```

You can include this statement at the following hierarchy levels:

- `[edit interfaces interface-name]`
- `[edit logical-systems logical-system-name interfaces interface-name]`

By default, the tunnel destination address is assumed to be in the default Internet routing table, inet.0. For IPsec tunnels using manual security association (SA), if the tunnel destination address is not in the default inet.0 routing table, you need to specify which routing table to search for the tunnel destination address by configuring the `routing-instance` statement. This is the case if the tunnel encapsulating interface is also configured under the routing instance.

```
unit logical-unit-number {
    tunnel {
        source address;
        destination address;
        routing-instance {
            destination routing-instance-name;
        }
        family inet {
            address address;
            ipsec-sa security-association-name;
        }
        family mpls;
    }
}
```

You can include these statements at the following hierarchy levels:

- `[edit interfaces interface-name]`
- `[edit logical-systems logical-system-name interfaces interface-name]`

NOTE: For IPsec tunnels using dynamic SA, the tunnel destination address must be in the default Internet routing table, inet.0.

To complete the ES tunnel interface configuration, include the `interface` statement for the ES interface under the appropriate routing instance:

```
interface interface-name;
```

You can include this statement at the following hierarchy levels:

- `[edit routing-instances routing-instance-name]`
- `[edit logical-systems logical-system-name routing-instances routing-instance-name]`

Configuring the ES Tunnel Interface on the CE Router

To configure the ES tunnel interface on the CE router, include the `unit` statement:

```
unit 0 {
    tunnel {
        source address;
        destination address;
    }
    family inet {
        address address;
        ipsec-sa security-association-name;
    }
}
```

You can include this statement at the following hierarchy levels:

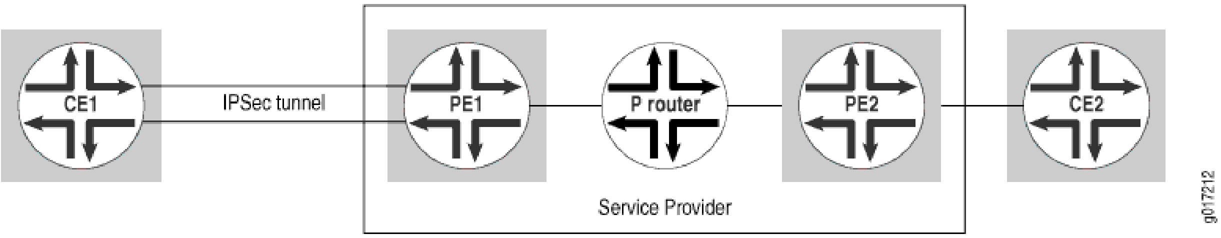
- `[edit interfaces interface-name]`
- `[edit logical-systems logical-system-name interfaces interface-name]`

Configuring an ES Tunnel Interface Between a PE and CE Router

IN THIS SECTION

This example shows how to configure an ES tunnel interface between a PE router and a CE router in a Layer 3 VPN. The network topology used in this example is shown in Figure 68 on page 875.

Figure 68: ES Tunnel Interface (IPsec Tunnel)



To configure this example, you perform the steps in the following sections:

Configuring IPsec on Router PE1

Configure IP Security (IPsec) on Router PE1:

```
[edit security]
ipsec {
    security-association sa-esp-manual {
        mode tunnel;
        manual {
            direction bidirectional {
                protocol esp;
                spi 16000;
                authentication {
```

```
                    algorithm hmac-md5-96;
                    key ascii-text "$9$ABULt1heK87dsWLDk.P3nrevM7V24ZHkPaZ/tp0cSvWLNwgZUH";
                }
                encryption {
                    algorithm des-cbc;
                    key ascii-text "$9$/H8Q90IyrvL7VKMZjHqQzcyleLN";
                }
            }
        }
    }
}
```

Configuring the Routing Instance Without the Encapsulating Interface

You can configure the routing instance on Router PE1 with or without the encapsulating interface (`t3-0/1/3` in this example). The following sections explain how to configure the routing instance without it:

Configuring the Routing Instance on Router PE1

Configure the routing instance on Router PE1:

```
[edit routing-instances]
vpna {
    instance-type vrf;
    interface es-1/2/0.0;
    route-distinguisher 10.255.14.174:1;
    vrf-import vpna-import;
    vrf-export vpna-export;
    protocols {
        bgp {
            group vpna {
                type external;
                peer-as 100;
                as-override;
                neighbor 10.49.2.1;
            }
        }
    }
}
```

Configuring the ES Tunnel Interface on Router PE1

Configure the ES tunnel interface on Router PE1:

```
[edit interfaces es-1/2/0]
unit 0 {
    tunnel {
        source 192.168.197.249;
        destination 192.168.197.250;
    }
    family inet {
        address 10.49.2.2/30;
        ipsec-sa sa-esp-manual;
    }
}
```

Configuring the Encapsulating Interface for the ES Tunnel

For this example, interface `t3-0/1/3` is the encapsulating interface for the ES tunnel. Configure interface `t3-0/1/3`:

```
[edit interfaces t3-0/1/3]
unit 0 {
    family inet {
        address 192.168.197.249/30;
    }
}
```

Configuring the Routing Instance with the Encapsulating Interface

If the tunnel-encapsulating interface, `t3-0/1/3`, is also configured under the routing instance, you need to specify the routing instance name under the interface definition. The system uses this routing instance to search for the tunnel destination address for the IPsec tunnel using manual security association.

The following sections explain how to configure the routing instance with the encapsulating interface:

Configuring the Routing Instance on Router PE1

Configure the routing instance on Router PE1 (including the tunnel encapsulating interface):

```
[edit routing-instances]
vpna {
    instance-type vrf;
    interface es-1/2/0.0;
    interface t3-0/1/3.0;
    route-distinguisher 10.255.14.174:1;
    vrf-import vpna-import;
    vrf-export vpna-export;
    protocols {
        bgp {
            group vpna {
                type external;
                peer-as 100;
                as-override;
                neighbor 10.49.2.1;
            }
        }
    }
}
```

Configuring the ES Tunnel Interface on Router PE1

Configure the ES tunnel interface on Router PE1:

```
[edit interfaces es-1/2/0]
unit 0 {
    tunnel {
        source 192.168.197.249;
        destination 192.168.197.250;
        routing-instance {
            destination vpna;
        }
    }
    family inet {
        address 10.49.2.2/30;
        ipsec-sa sa-esp-manual;
```

```
    }
}
```

Configuring the Encapsulating Interface on Router PE1

Configure the encapsulating interface on Router PE1:

```
[edit interfaces t3-0/1/3]
unit 0 {
    family inet {
        address 192.168.197.249/30;
    }
}
```

Configuring the ES Tunnel Interface on Router CE1

Configure the ES tunnel interface on Router CE1:

```
[edit interfaces es-1/2/0]
unit 0 {
    tunnel {
        source 192.168.197.250;
        destination 192.168.197.249;
    }
    family inet {
        address 10.49.2.1/30;
        ipsec-sa sa-esp-manual;
    }
}
```

Configuring IPsec on Router CE1

Configure IPsec on Router CE1:

```
[edit security]
ipsec {
    security-association sa-esp-manual {
        mode tunnel;
```

```
        manual {
            direction bidirectional {
                protocol esp;
                spi 16000;
                authentication {
                    algorithm hmac-md5-96;
                    key ascii-text "$9$ABULt1heK87dsWLDk.P3nrevM7V24ZHkPaZ/tp0cSvWLNwgZUH";
                }
                encryption {
                    algorithm des-cbc;
                    key ascii-text "$9$/H8Q90IyrvL7VKMZjHqQzcyleLN";
                }
            }
        }
    }
}
```

GRE Tunnels for Layer 3 VPNs

IN THIS SECTION

Configuring GRE Tunnels for Layer 3 VPNs

IN THIS SECTION

Junos OS allows you to configure a generic routing encapsulation (GRE) tunnel between the PE and CE routers for a Layer 3 VPN. The GRE tunnel can have one or more hops. You can configure the tunnel from the PE router to a local CE router (as shown in Figure 69 on page 881) or to a remote CE router (as shown in Figure 70 on page 881).

Figure 69: GRE Tunnel Configured Between the Local CE Router and the PE Router

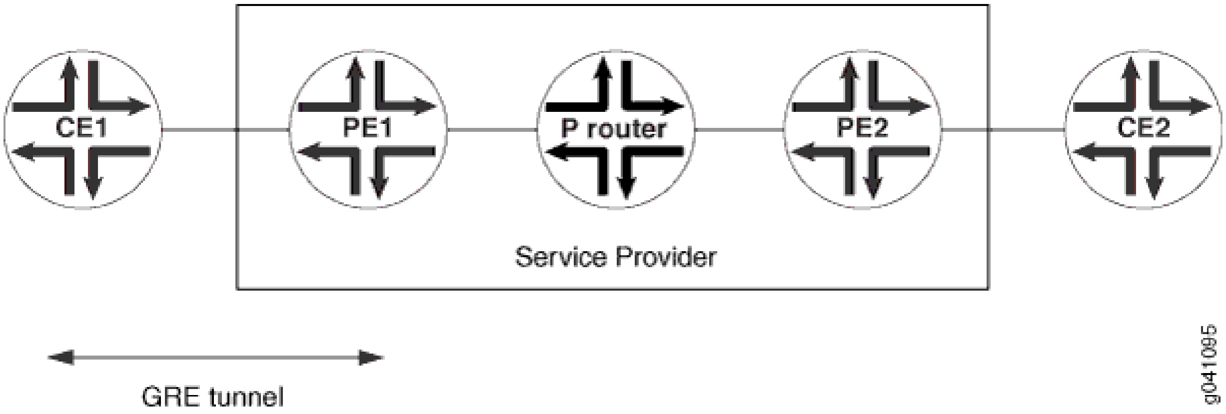
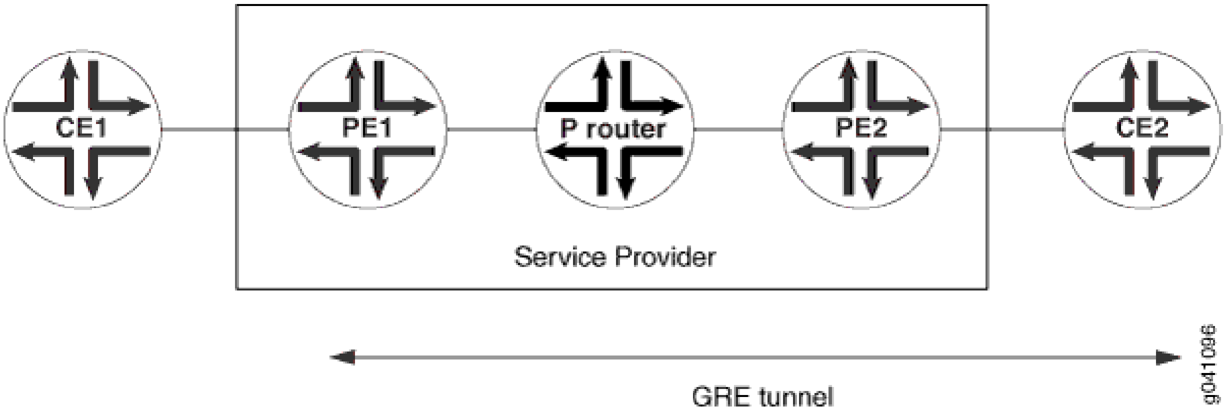


Figure 70: GRE Tunnel Configured Between the Remote CE Router and the PE Router



For more information about how to configure tunnel interfaces, see the Junos OS Services Interfaces Library for Routing Devices.

You can configure the GRE tunnels manually or configure the Junos OS to instantiate GRE tunnels dynamically.

The following sections describe how to configure GRE tunnels manually and dynamically:

Configuring GRE Tunnels Manually Between PE and CE Routers

You can manually configure a GRE tunnel between a PE router and either a local CE router or a remote CE router for a Layer 3 VPN as explained in the following sections:

Configuring the GRE Tunnel Interface on the PE Router

You configure the GRE tunnel as a logical interface on the PE router. To configure the GRE tunnel interface, include the `unit` statement:

```
unit logical-unit-number {
    tunnel {
        source source-address;
        destination destination-address;
        routing-instance {
            destination routing-instance-name;
        }
    }
    family inet {
        address address;
    }
}
```

You can include this statement at the following hierarchy levels:

- `[edit interfaces` *`interface-name`*`]`
- `[edit logical-systems` *`logical-system-name`* `interfaces` *`interface-name`*`]`

As part of the GRE tunnel interface configuration, you need to include the following statements:

- `source` *`source-address`*—Specify the source or origin of the GRE tunnel, typically the PE router.
- `destination` *`destination-address`*—Specify the destination or end point of the GRE tunnel. The destination can be a Provider router, the local CE router, or the remote CE router.

By default, the tunnel destination address is assumed to be in the default Internet routing table, inet.0. If the tunnel destination address is not in inet.0, you need to specify which routing table to search for the tunnel destination address by configuring the `routing-instance` statement. This is the case if the tunnel encapsulating interface is also configured under the routing instance.

- `destination` *`routing-instance-name`*—Specify the name of the routing instance when configuring the GRE tunnel interface on the PE router.

To complete the GRE tunnel interface configuration, include the `interface` statement for the GRE interface under the appropriate routing instance:

```
interface interface-name;
```

You can include this statement at the following hierarchy levels:

- `[edit routing-instances routing-instance-name]`
- `[edit logical-systems logical-system-name routing-instances routing-instance-name]`

Configuring the GRE Tunnel Interface on the CE Router

You can configure either the local or the remote CE router to act as the endpoint for the GRE tunnel.

To configure the GRE tunnel interface on the CE router, include the `unit` statement:

```
unit logical-unit-number {
    tunnel {
        source address;
        destination address;
    }
    family inet {
        address address;
    }
}
```

You can include this statement at the following hierarchy levels:

- `[edit interfaces interface-name]`
- `[edit logical-systems logical-system-name interfaces interface-name]`

Configuring GRE Tunnels Dynamically

When the router receives a VPN route to a BGP next hop address, but no MPLS path is available, a GRE tunnel can be dynamically generated to carry the VPN traffic across the BGP network. The GRE tunnel is generated and then its routing information is copied into the inet.3 routing table. IPv4 routes are the only type of routes supported for dynamic GRE tunnels. Also, the routing platform must have a tunnel PIC.

NOTE: When configuring a dynamic GRE tunnel to a remote CE router, do not configure OSPF over the tunnel interface. It creates a routing loop forcing the router to take the GRE tunnel down. The router attempts to reestablish the GRE tunnel, but will be forced to take it down again when OSPF becomes active on the tunnel interface and discovers a route to the tunnel endpoint. This is not an issue when configuring static GRE tunnels to a remote CE router.

To generate GRE tunnels dynamically, include the `dynamic-tunnels` statement:

```
dynamic-tunnels tunnel-name {
    destination-networks prefix;
    source-address address;
}
```

You can include this statement at the following hierarchy levels:

- `[edit routing-options]`
- `[edit routing-instances` *`routing-instance-name`* `routing-options]`
- `[edit logical-systems` *`logical-system-name`* `routing-options]`
- `[edit logical-systems` *`logical-system-name`* `routing-instances` *`routing-instance-name`* `routing-options ]`

Specify the IPv4 prefix range (for example, 10/8 or 11.1/16) for the destination network by including the `destination-networks` statement. Only tunnels within the specified IPv4 prefix range are allowed to be initiated.

```
destination-networks prefix;
```

You can include this statement at the following hierarchy levels:

- `[edit routing-options]`
- `[edit routing-instances` *`routing-instance-name`* `routing-options]`
- `[edit logical-systems` *`logical-system-name`* `routing-options]`
- `[edit logical-systems` *`logical-system-name`* `routing-instances` *`routing-instance-name`* `routing-options ]`

Specify the source address for the GRE tunnels by including the `source-address` statement. The source address specifies the address used as the source for the local tunnel endpoint. This could be any local address on the router (typically the router ID or the loopback address).

```
source-address address;
```

You can include this statement at the following hierarchy levels:

- `[edit routing-options]`
- `[edit routing-instances routing-instance-name routing-options]`
- `[edit logical-systems logical-system-name routing-options]`
- `[edit logical-systems logical-system-name routing-instances routing-instance-name routing-options ]`

SEE ALSO

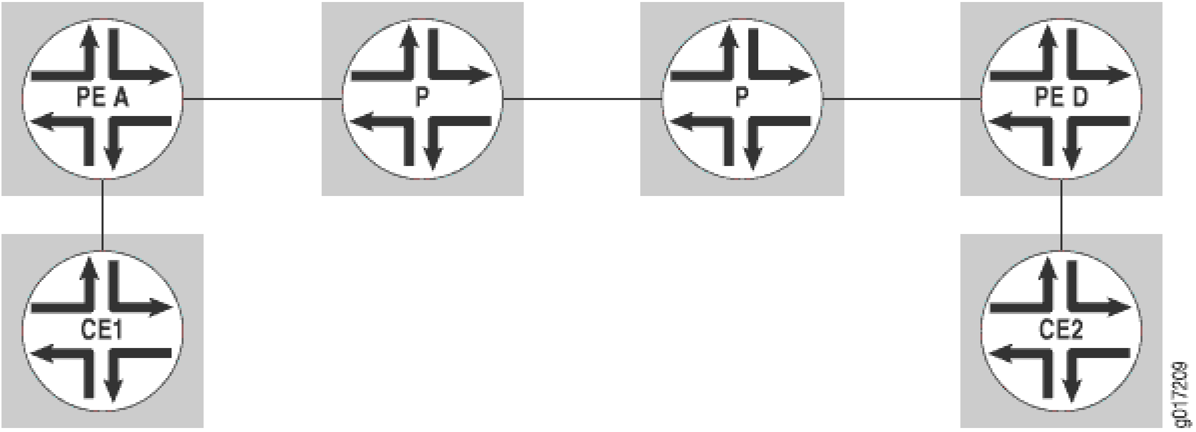
Example: Configuring a Two-Tiered Virtualized Data Center for Large Enterprise Networks

Configuring a GRE Tunnel Between PE Routers

IN THIS SECTION

This example shows how to configure a generic routing encapsulation (GRE) tunnel interface between PE routers to provide VPN connectivity. You can use this configuration to tunnel VPN traffic across a non-MPLS core network. The network topology used in this example is shown in Figure 71 on page 886. The P routers shown in this illustration do not run MPLS.

Figure 71: PE Routers A and D Connected by a GRE Tunnel Interface



For configuration information, see the following sections:

Configuring the Routing Instance on Router A

Configure a routing instance on Router A:

```
[edit routing-instances]
gre-config {
    instance-type vrf;
    interface fe-1/0/0.0;
    route-distinguisher 10.255.14.176:69;
    vrf-import import-config;
    vrf-export export-config;
    protocols {
        ospf {
            export import-config;
            area 0.0.0.0 {
                interface all;
            }
        }
    }
}
```

Configuring the Routing Instance on Router D

Configure a routing instance on Router D:

```
[edit routing-instances]
gre-config {
    instance-type vrf;
    interface fe-1/0/1.0;
    route-distinguisher 10.255.14.178:69;
    vrf-import import-config;
    vrf-export export-config;
    protocols {
        ospf {
            export import-config;
            area 0.0.0.0 {
                interface all;
            }
        }
    }
}
```

Configuring MPLS, BGP, and OSPF on Router A

Although you do not need to configure MPLS on the P routers in this example, it is needed on the PE routers for the interface between the PE and CE routers and on the GRE interface (`gr-1/1/0.0`) linking the PE routers (Router A and Router D). Configure MPLS, BGP, and OSPF on Router A:

```
[edit protocols]
mpls {
    interface all;
}
bgp {
    group pe-to-pe {
        type internal;
        neighbor 10.255.14.178 {
            family inet-vpn {
                unicast;
            }
        }
    }
}
```

```
ospf {
    area 0.0.0.0 {
        interface all;
        interface gr-1/1/0.0 {
            disable;
        }
    }
}
```

Configuring MPLS, BGP, and OSPF on Router D

Although you do not need to configure MPLS on the P routers in this example, it is needed on the PE routers for the interface between the PE and CE routers and on the GRE interface (`gr-1/1/0.0`) linking the PE routers (Router D and Router A). Configure MPLS, BGP, and OSPF on Router D:

```
[edit protocols]
mpls {
    interface all;
}
bgp {
    group pe-to-pe {
        type internal;
        neighbor 10.255.14.176 {
            family inet-vpn {
                unicast;
            }
        }
    }
}
ospf {
    traffic-engineering;
    area 0.0.0.0 {
        interface all;
        interface fxp0.0 {
            disable;
        }
        interface gr-1/1/0.0 {
            disable;
        }
    }
}
```

Configuring the Tunnel Interface on Router A

Configure the tunnel interface on Router A (the tunnel is unnumbered):

```
[edit interfaces interface-name]
unit 0 {
    tunnel {
        source 10.255.14.176;
        destination 10.255.14.178;
    }
    family inet;
    family mpls;
}
```

Configuring the Tunnel Interface on Router D

Configure the tunnel interface on Router D (the tunnel is unnumbered):

```
[edit interfaces interface-name]
unit 0 {
    tunnel {
        source 10.255.14.178;
        destination 10.255.14.176;
    }
    family inet;
    family mpls;
}
```

Configuring the Routing Options on Router A

As part of the routing options configuration for Router A, you need to configure routing table groups to enable VPN route resolution in the inet.3 routing table.

Configure the routing options on Router A:

```
[edit routing-options]
interface-routes {
    rib-group inet if-rib;
}
rib inet.3 {
```

```
    static {
        route 10.255.14.178/32 next-hop gr-1/1/0.0;
    }
}
rib-groups {
    if-rib {
        import-rib [ inet.0 inet.3 ];
    }
}
```

Configuring the Routing Options on Router D

As part of the routing options configuration for Router D, you need to configure routing table groups to enable VPN route resolution in the inet.3 routing table.

Configure the routing options on Router D:

```
[edit routing-options]
interface-routes {
    rib-group inet if-rib;
}
rib inet.3 {
    static {
        route 10.255.14.176/32 next-hop gr-1/1/0.0;
    }
}
rib-groups {
    if-rib {
        import-rib [ inet.0 inet.3 ];
    }
}
```

Configuration Summary for Router A

Configure the Routing Instance

```
gre-config {
    instance-type vrf;
    interface fe-1/0/0.0;
    route-distinguisher 10.255.14.176:69;
```

```
    vrf-import import-config;
    vrf-export export-config;
    protocols {
        ospf {
            export import-config;
            area 0.0.0.0 {
                interface all;
            }
        }
    }
}
```

Configure MPLS

```
mpls {
    interface all;
}
```

Configure BGP

```
bgp {
    traceoptions {
        file bgp.trace world-readable;
        flag update detail;
    }
    group pe-to-pe {
        type internal;
        neighbor 10.255.14.178 {
            family inet-vpn {
                unicast;
            }
        }
    }
}
```

Configure OSPF

```
ospf {
    area 0.0.0.0 {
        interface all;
        interface gr-1/1/0.0 {
```

```
            disable;
        }
    }
}
```

Configure the Tunnel Interface

```
interface-name {
    unit 0 {
        tunnel {
            source 10.255.14.176;
            destination 10.255.14.178;
        }
        family inet;
        family mpls;
    }
}
```

Configure Routing Options

```
interface-routes {
    rib-group inet if-rib;
}
rib inet.3 {
    static {
        route 10.255.14.178/32 next-hop gr-1/1/0.0;
    }
}
rib-groups {
    if-rib {
        import-rib [ inet.0 inet.3 ];
    }
}
```

Configuration Summary for Router D

Configure the Routing Instance

```
gre-config {
    instance-type vrf;
```

```
    interface fe-1/0/1.0;
    route-distinguisher 10.255.14.178:69;
    vrf-import import-config;
    vrf-export export-config;
    protocols {
        ospf {
            export import-config;
            area 0.0.0.0 {
                interface all;
            }
        }
    }
}
```

Configure MPLS

```
mpls {
    interface all;
}
```

Configure BGP

```
bgp {
    group pe-to-pe {
        type internal;
        neighbor 10.255.14.176 {
            family inet-vpn {
                unicast;
            }
        }
    }
}
```

Configure OSPF

```
ospf {
    traffic-engineering;
    area 0.0.0.0 {
        interface all;
        interface fxp0.0 {
            disable;
```

```
        }
        interface gr-1/1/0.0 {
            disable;
        }
    }
}
```

Configure the Tunnel Interface

```
interface-name {
    unit 0 {
        tunnel {
            source 10.255.14.178;
            destination 10.255.14.176;
        }
        family inet;
        family mpls;
    }
}
```

Configure the Routing Options

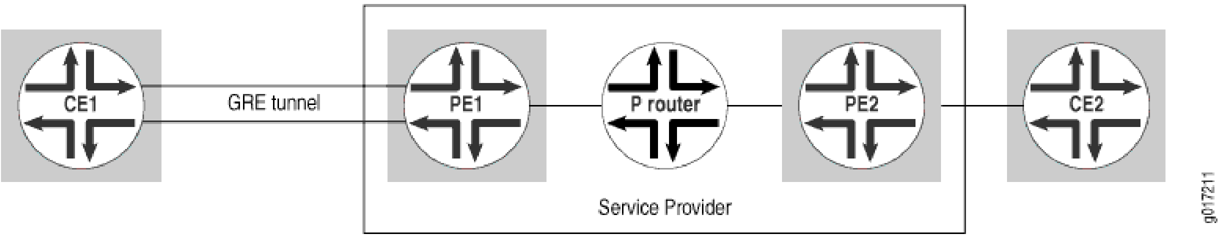
```
interface-routes {
    rib-group inet if-rib;
}
rib inet.3 {
    static {
        route 10.255.14.176/32 next-hop gr-1/1/0.0;
    }
}
rib-groups {
    if-rib {
        import-rib [ inet.0 inet.3 ];
    }
}
```

Configuring a GRE Tunnel Interface Between a PE and CE Router

IN THIS SECTION

This example shows how to configure a GRE tunnel interface between a PE router and a CE router. You can use this configuration to tunnel VPN traffic across a non-MPLS core network. The network topology used in this example is shown in Figure 72 on page 895.

Figure 72: GRE Tunnel Between the CE Router and the PE Router



For this example, complete the procedures described in the following sections:

Configuring the Routing Instance Without the Encapsulating Interface

You can configure the routing instance either with or without the encapsulating interface. The following sections explain how to configure the routing instance without it:

Configuring the Routing Instance on Router PE1

Configure the routing instance on Router PE1:

```
[edit routing-instances]
vpna {
    instance-type vrf;
    interface gr-1/2/0.0;
    route-distinguisher 10.255.14.174:1;
    vrf-import vpna-import;
```

```
    vrf-export vpna-export;
    protocols {
        bgp {
            group vpna {
                type external;
                peer-as 100;
                as-override;
                neighbor 10.49.2.1;
            }
        }
    }
}
```

Configuring the GRE Tunnel Interface on Router PE1

Configure the GRE tunnel interface on Router PE1:

```
[edit interfaces gr-1/2/0]
unit 0 {
    tunnel {
        source 192.168.197.249;
        destination 192.168.197.250;
    }
    family inet {
        address 10.49.2.2/30;
    }
}
```

In this example, interface `t3-0/1/3` acts as the encapsulating interface for the GRE tunnel.

When you configure the `clear-dont-fragment-bit` statement on an interface with the MPLS protocol family enabled, you must specify an MTU value. This MTU value must not be greater than the maximum supported value, which is 9192.

For example:

```
user@host# show interfaces gr-1/2/0
unit 0 {
    clear-dont-fragment-bit;
    family inet {
        mtu 9100;
```

```
        address 10.10.1.1/32;
    }
    family mpls {
        mtu 9100;
    }
}
```

Configuring the Encapsulation Interface on Router PE1

Configure the encapsulation interface on Router PE1:

```
[edit interfaces t3-0/1/3]
unit 0 {
    family inet {
        address 192.168.197.249/30;
    }
}
```

Configuring the Routing Instance with the Encapsulating Interface

If the tunnel-encapsulating interface, `t3-0/1/3`, is also configured under the routing instance, then you need to specify the name of that routing instance under the interface definition. The system uses this routing instance to search for the tunnel destination address.

To configure the routing instance with the encapsulating interface, you perform the steps in the following sections:

Configuring the Routing Instance on Router PE1

If you configure the tunnel-encapsulating interface under the routing instance, then configure the routing instance on Router PE1:

```
[edit routing-instances]
vpna {
    instance-type vrf;
    interface gr-1/2/0.0;
    interface t3-0/1/3.0;
    route-distinguisher 10.255.14.174:1;
    vrf-import vpna-import;
    vrf-export vpna-export;
```

```
    protocols {
        bgp {
            group vpna {
                type external;
                peer-as 100;
                as-override;
                neighbor 10.49.2.1;
            }
        }
    }
}
```

Configuring the GRE Tunnel Interface on Router PE1

Configure the GRE tunnel interface on Router PE1:

```
[edit interfaces gr-1/2/0]
unit 0 {
    tunnel {
        source 192.168.197.249;
        destination 192.168.197.250;
        routing-instance {
            destination vpna;
        }
    }
    family inet {
        address 10.49.2.2/30;
    }
}
```

When you configure the `clear-dont-fragment-bit` statement on an interface with the MPLS protocol family enabled, you must specify an MTU value. This MTU value must not be greater than the maximum supported value, which is 9192.

For example:

```
user@host# show interfaces gr-1/2/0
unit 0 {
    clear-dont-fragment-bit;
    family inet {
        mtu 9100;
```

```
        address 10.10.1.1/32;
    }
    family mpls {
        mtu 9100;
    }
}
```

When you configure the `clear-dont-fragment-bit` statement on an interface with the MPLS protocol family enabled, you must specify an MTU value. This MTU value must not be greater than the maximum supported value, which is 9192.

For example:

```
user@host# show interfaces gr-1/2/0
unit 0 {
    clear-dont-fragment-bit;
    family inet {
        mtu 9100;
        address 10.10.1.1/32;
    }
    family mpls {
        mtu 9100;
    }
}
```

Configuring the Encapsulation Interface on Router PE1

Configure the encapsulation interface on Router PE1:

```
[edit interfaces t3-0/1/3]
unit 0 {
    family inet {
        address 192.168.197.249/30;
    }
}
```

Configuring the GRE Tunnel Interface on Router CE1

Configure the GRE tunnel interface on Router CE1:

```
[edit interfaces gr-1/2/0]
unit 0 {
    tunnel {
        source 192.168.197.250;
        destination 192.168.197.249;
    }
    family inet {
        address 10.49.2.1/30;
    }
}
```

Next-Hop Based Tunnels for Layer 3 VPNs

IN THIS SECTION

This topic describes configuring dynamic generic routing encapsulation (GRE) tunnel and a dynamic MPLS-over-UDP tunnel to support tunnel composite next hop. It also provides information on configuring reverse path forwarding to protect against anti-spoofing.

Configure Next-hop-based MPLS-over-GRE Dynamic Tunnels

IN THIS SECTION

Next-hop-based MPLS-over-GRE tunnels creates a tunnel composite next hop, an indirect next hop, and a forwarding next hop to resolve a tunnel destination route. You can configure next-hop-based MPLS-over-GRE tunnels along with firewall filter-based tunnel decapsulation.

On ACX Series, starting from Junos OS Evolved Release 24.2R1, you can configure dynamic GRE next-hop-based tunnel by including the `gre next-hop-based-tunnel` statement at the [`edit routing-options dynamic-tunnels`] hierarchy.

You can configure MPLS-over-GRE firewall filter-based decapsulation by including the `gre` statement at the [`edit firewall family family-name filter filter-name term term-name then decapsulate`] hierarchy level. Only decapsulate action is supported in the firewall filter rule. MPLS-over-GRE firewall filter-based decapsulation is supported only for family inet.

You can retrieve encapsulation tunnel statistics by configuring specific interval using the `interval` statement at the [`edit routing-options dynamic-tunnels statistics`] hierarchy level. You can also use the `show dynamic-tunnels database statistics` command to view the statistics.

Configure Encapsulation for Dynamic Next-hop Tunnel

To configure encapsulation for dynamic next-hop tunnel, you need to include the `gre next-hop-based-tunnel` statement at the [`edit routing-options dynamic-tunnels`] hierarchy level.

The following is a sample configuration for MPLS-over-GRE Encapsulation:

```
[edit]
protocols {
    bgp {
        group IBGP {
            type internal;
```

```
            local-address 192.168.19.190;
            family inet {
                unicast;
           }
       }
       neighbor 192.168.20.11;
    }
}
routing-options {
    dynamic-tunnels {
        gre full-resolved-next-hop-based-tunnel;
        gre next-hop-based-tunnel;
        test_tunnel {
            source-address 192.168.19.190;
            gre;
            destination-networks {
                192.168.20.0/24 preference 4;
            }
        }
    }
}
```

Configure Firewall Filters for Decapsulation

In the following configuration, we match the SIP, DIP, protocol to match tunnel packets. Firewall filter needs to be mapped to the default routing instance.

With the action, PFE programs the router to strip off the tunnel headers and continue processing based on the inner header.

```

forwarding-options {
    family inet {
        filter {
            input f1;
        }
    }
}
firewall {
    family inet {
        filter f1 {
            term t1 {
```

```
                from {
                    source-address {
                        192.168.9.190/32;
                    }
                    destination-address {
                        10.255.10.11/32;
                    }                }
                then {
                    decapsulate {
                        gre;
                    }
                }
            }
            term t2 {
                then accept;
            }
        }
    }
}
```

Enabling Encapsulation Statistics

PFE has the capability to provide the encapsulation statistics at a packet and at a byte count level. The byte count value includes the encapsulation header. By default, the statistics are not enabled for the tunnel encapsulation as the counters used for statistics is a shared resource. Therefore, you need to enable counter allocation by configuring the `encap-stats-enable` CLI statement.

To enable the encapsulation statistics, configure the following CLI:

```
user@host# set system packet-forwarding-options tunnel encap-stats-enable
```

Limitations

The following are the limitations while configuring next hop-based MPLS-over-GRE Dynamic Tunnels on ACX Series:

- TTL propagation for the dynamic next hop-based tunnel is not supported.
- Class of service and MTU is not supported.

- Path MTU discovery for both IPv4 and IPv6 encapsulation is not supported because flow cache is not maintained for the encapsulated packet at tunnel originating node.
- Anti-spoofing at tunnel decapsulation for the inner source IP is not supported.
- In a firewall filter rule, only `decapsulate mpls-in-udp` and `decapsulate gre` is supported for tunnel decapsulation. Other firewall filter rules over the incoming core interface is not supported due to data path limitation.
- Firewall filter decapsulation rule supports only the default VRF.
- Tunnel statistics per tunnel composite next hop level is not supported.
- Statistics counter allocation is on a first come first serve (FCFS) basis. Upon a system restart or evo-pefmand restart, some of the counters might have a non-zero value post restart and the counters might not increment.
- Prefix mask of /32 for IPv4 and /128 for IPv6 is only supported in decapsulation-based filter rules. Firewall filter prefix list in decapsulation-based filter rules is not supported.
- Flexible-vlan-tagging is not supported.
- Interface-based dynamic next-hop GRE tunnel is not supported.
- The `set chassis loopback-dynamic-tunnel` configuration statement is not applicable for the ACX platform because line-rate tunnel encapsulation and decapsulation are supported.
- IPv6 core is not supported.

Example: Configuring Next-Hop-Based Dynamic GRE Tunnels

IN THIS SECTION

This example shows how to configure a dynamic generic routing encapsulation (GRE) tunnel that includes a tunnel composite next hop (CNH) instead of an interface next hop. The next-hop-based dynamic GRE tunnel has a scaling advantage over the interface-based dynamic GRE tunnels.

Requirements

This example uses the following hardware and software components:

- Five MX Series routers with MPCs and MICs.
- Junos OS Release 16.2 or later running on the PE routers.

Before you begin:

1. Configure the device interfaces, including the loopback interface.
2. Configure the router ID and autonomous system number for the device.
3. Establish an internal BGP (IBGP) session with the remote PE device.
4. Establish OSPF peering among the devices.

Overview

IN THIS SECTION

Starting with Junos OS Release 16.2, a dynamic generic routing encapsulation (GRE) tunnel supports the creation of a tunnel composite next hop for every GRE tunnel configured.

By default, for every new dynamic GRE tunnel configured, a corresponding logical tunnel interface is created. This is called an interface-based dynamic tunnel and is the default mode for creating dynamic tunnels. On interface-based dynamic tunnel, the number of tunnels that can be configured on a device depends on the total number of interfaces supported on the device. The next-hop-based dynamic GRE tunnel removes the dependency on physical interfaces, and the GRE tunnel encapsulation is implemented as a next-hop instruction without creating a logical tunnel interface. This provides a scaling advantage for the number of dynamic tunnels that can be created on a device.

Starting in Junos OS Release 17.1, on MX Series routers with MPCs and MICs, the scaling limit of next-hop-based dynamic GRE tunnels is increased. The increased scaling values benefits data center networks, where a gateway router is required to communicate with a number of servers over an IP infrastructure; for example, in Contrail Networking.

At a given point in time, either the next-hop-based dynamic tunnel or the default interface-based dynamic GRE tunnel can exist on a device. A switchover from one tunnel mode to another deletes the existing tunnels and creates new tunnels in the new tunnel mode according to the supported scaling limits. Similarly, at a given point in time, for the same tunnel destination, the next-hop-based tunnel encapsulation type can either be GRE or UDP.

To enable the next-hop-based dynamic GRE tunnel, include the `next-hop-based-tunnel` statement at the `[edit routing-options dynamic-tunnels gre]` hierarchy level.

The existing dynamic tunnel feature requires complete static configuration. Currently, the tunnel information received from peer devices in advertised routes is ignored. Starting in Junos OS Release 17.4R1, on MX Series routers, the next-hop-based dynamic GRE tunnels are signaled using BGP encapsulation extended community. BGP export policy is used to specify the tunnel types, advertise the sender side tunnel information, and parse and convey the receiver side tunnel information. A tunnel is created according to the received type tunnel community.

Multiple tunnel encapsulations are supported by BGP. On receiving multiple capability, the next-hop-based dynamic tunnel is created based on the configured BGP policy and tunnel preference. The tunnel preference should be consistent across both the tunnel ends for the tunnel to be set up. By default, MPLS-over-UDP (MPLSoUDP) tunnel is preferred over GRE tunnels. If dynamic tunnel configuration exists, it takes precedence over received tunnel community.

When configuring a next-hop-based dynamic GRE tunnel, be aware of the following considerations:

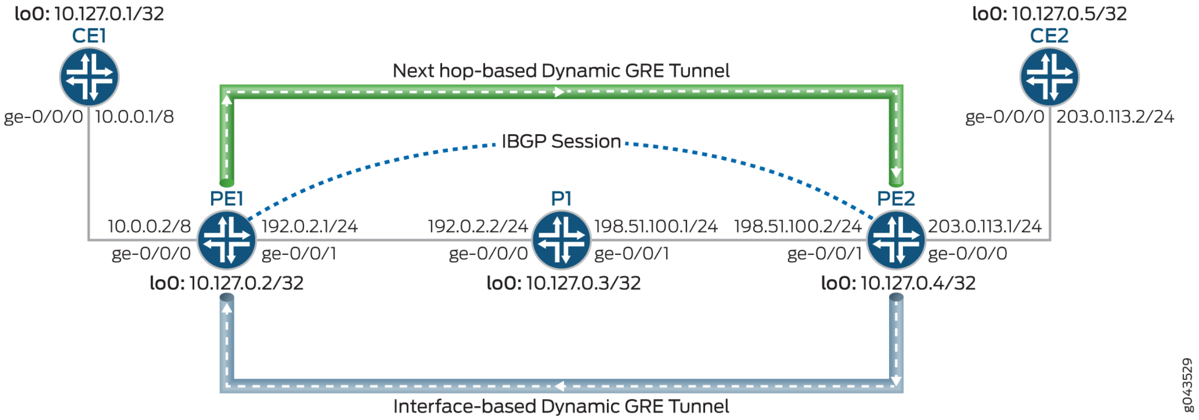
- Dynamic tunnels creation is triggered in the IBGP protocol next-hop resolution process.
- A switchover from the next-hop-based tunnel mode to the interface-based tunnel mode (and vice versa) is allowed, and this can impact network performance in terms of the supported IP tunnel scaling values in each mode.
- RSVP automatic mesh tunnel is not supported with the next-hop-based dynamic tunnel configuration.
- Dynamic GRE tunnel creation based upon new IPv4-mapped-IPv6 next hops is supported with this feature.
- The following features are not supported with the next-hop-based dynamic GRE tunnel configuration:
  - RSVP automatic mesh
  - Logical systems
  - Per-tunnel traffic statistics collection on the sender (MX Series) side
  - QoS enforcement, such as policing and shaping, at the tunnel interface.
  - Path maximum transmission unit discovery
  - GRE key feature

- GRE Operation, Administration, and Maintenance (OAM) for GRE keepalive messages.

Topology

Figure 73 on page 907 illustrates a Layer 3 VPN scenario over dynamic GRE tunnels. The customer edge (CE) devices, CE1 and CE2, connect to provider edge (PE) devices, PE1 and PE2, respectively. The PE devices are connected to a provider device (Device P1), and an internal BGP (IBGP) session interconnects the two PE devices. Two dynamic GRE tunnels are configured between the PE devices. The dynamic tunnel from Device PE1 to Device PE2 is based on the next hop tunnel mode, and the dynamic tunnel from Device PE2 to Device PE1 is based on the interface tunnel mode.

Figure 73: Layer 3 VPN over Dynamic GRE Tunnels



The next-hop-based dynamic GRE tunnel is handled as follows:

1. After a next-hop-based dynamic GRE tunnel is configured, a tunnel destination mask route with a tunnel CNH is created for the protocol next hops in the inet.3 routing table. This IP tunnel route is withdrawn only when the dynamic tunnel configuration is deleted.

   The tunnel CNH attributes include the following:

   - When Layer 3 VPN CNH is disabled—Source and destination address, encapsulation string, and VPN label.
   - When Layer 3 VPN CNH and per-prefix VPN label allocation are enabled—Source address, destination address, and encapsulation string.
   - When Layer 3 VPN CNH is enabled and per-prefix VPN label allocation is disabled—Source address, destination address, and encapsulation string. The route in this case is added to the other virtual routing and forwarding instance table with a secondary route.

2. The PE devices are interconnected using an IBGP session. The IBGP route next hop to a remote BGP neighbor is the protocol next hop, which is resolved using the tunnel mask route with the tunnel next hop.
3. After the next hop is resolved over the tunnel composite next hop, indirect next hops with forwarding next hops are created.
4. The tunnel CNH is used to forward the next hops of the indirect next hops.

Configuration

IN THIS SECTION

CLI Quick Configuration

To quickly configure this example, copy the following commands, paste them into a text file, remove any line breaks, change any details necessary to match your network configuration, copy and paste the commands into the CLI at the `[edit]` hierarchy level, and then enter `commit` from configuration mode.

CE1

```
set interfaces ge-0/0/0 unit 0 family inet address 10.0.0.1/8
set interfaces lo0 unit 0 family inet address 10.127.0.1/8
set routing-options router-id 10.127.0.1
set routing-options autonomous-system 65200
set protocols bgp group ce1-pe1 export export-loopback-direct
set protocols bgp group ce1-pe1 peer-as 65100
set protocols bgp group ce1-pe1 neighbor 10.0.0.2
set policy-options policy-statement export-loopback-direct term term-1 from interface lo0.0
set policy-options policy-statement export-loopback-direct term term-1 from route-filter
10.127.0.1/8 exact
set policy-options policy-statement export-loopback-direct term term-1 then accept
```

CE2

```
set interfaces ge-0/0/0 unit 0 family inet address 203.0.113.2/24
set interfaces lo0 unit 0 family inet address 10.127.0.5/32
set routing-options router-id 10.127.0.5
set routing-options autonomous-system 65200
set protocols bgp group ce1-pe1 export export-loopback-direct
set protocols bgp group ce1-pe1 peer-as 65100
set protocols bgp group ce1-pe1 neighbor 203.0.113.1
set policy-options policy-statement export-loopback-direct term term-1 from interface lo0.0
set policy-options policy-statement export-loopback-direct term term-1 from route-filter
10.127.0.5/8 exact
set policy-options policy-statement export-loopback-direct term term-1 then accept
```

PE1

```
set interfaces ge-0/0/0 unit 0 family inet address 10.0.0.2/8
set interfaces ge-0/0/1 unit 0 family inet address 192.0.2.1/24
set interfaces ge-0/0/1 unit 0 family mpls
set interfaces lo0 unit 0 family inet address 10.127.0.2/32
set routing-options static route 10.33.0.0/16 next-hop 192.0.2.2
set routing-options router-id 10.127.0.2
set routing-options autonomous-system 65100
set routing-options dynamic-tunnels gre next-hop-based-tunnel
set routing-options dynamic-tunnels gre-dyn-tunnel-to-pe2 source-address 10.127.0.2
set routing-options dynamic-tunnels gre-dyn-tunnel-to-pe2 gre
set routing-options dynamic-tunnels gre-dyn-tunnel-to-pe2 destination-networks 10.127.0.0/16
set protocols bgp group IBGP type internal
set protocols bgp group IBGP local-address 10.127.0.2
set protocols bgp group IBGP family inet-vpn unicast
set protocols bgp group IBGP neighbor 10.127.0.4
set protocols ospf area 0.0.0.0 interface ge-0/0/1.0
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set routing-instances L3VPN-Over-GRE-PE1 instance-type vrf
set routing-instances L3VPN-Over-GRE-PE1 interface ge-0/0/0.0
set routing-instances L3VPN-Over-GRE-PE1 route-distinguisher 10.127.0.2:1
set routing-instances L3VPN-Over-GRE-PE1 vrf-target target:600:1
set routing-instances L3VPN-Over-GRE-PE1 protocols bgp group pe1-ce1 peer-as 65200
set routing-instances L3VPN-Over-GRE-PE1 protocols bgp group pe1-ce1 neighbor 10.0.0.1 as-
override
```

P1

```
set interfaces ge-0/0/0 unit 0 family inet address 192.0.2.2/24
set interfaces ge-0/0/0 unit 0 family mpls
set interfaces ge-0/0/1 unit 0 family inet address 198.51.100.1/24
set interfaces ge-0/0/1 unit 0 family mpls
set interfaces lo0 unit 0 family inet address 10.127.0.3/32
set routing-options router-id 10.127.0.3
set routing-options autonomous-system 65100
set protocols ospf area 0.0.0.0 interface ge-0/0/0.0
set protocols ospf area 0.0.0.0 interface ge-0/0/1.0
set protocols ospf area 0.0.0.0 interface lo0.0 passive
```

PE2

```
set interfaces ge-0/0/0 unit 0 family inet address 203.0.113.1/24
set interfaces ge-0/0/1 unit 0 family inet address 198.51.100.2/24
set interfaces lo0 unit 0 family inet address 10.127.0.4/32
set routing-options nonstop-routing
set routing-options router-id 10.127.0.4
set routing-options autonomous-system 65100
set routing-options dynamic-tunnels gre-dyn-tunnel-to-pe1 source-address 10.127.0.4
set routing-options dynamic-tunnels gre-dyn-tunnel-to-pe1 gre
set routing-options dynamic-tunnels gre-dyn-tunnel-to-pe1 destination-networks 10.127.0.0/16
set protocols bgp group IBGP type internal
set protocols bgp group IBGP local-address 10.127.0.4
set protocols bgp group IBGP family inet-vpn unicast
set protocols bgp group IBGP neighbor 10.127.0.2
set protocols ospf area 0.0.0.0 interface ge-0/0/1.0
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set routing-instances L3VPN-Over-GRE-PE2 instance-type vrf
set routing-instances L3VPN-Over-GRE-PE2 interface ge-0/0/0.0
set routing-instances L3VPN-Over-GRE-PE2 route-distinguisher 10.127.0.4:1
set routing-instances L3VPN-Over-GRE-PE2 vrf-target target:600:1
set routing-instances L3VPN-Over-GRE-PE2 protocols bgp group ebgp peer-as 65200
set routing-instances L3VPN-Over-GRE-PE2 protocols bgp group ebgp neighbor 203.0.113.2 as-
override
```

Procedure

Step-by-Step Procedure

The following example requires that you navigate various levels in the configuration hierarchy. For information about navigating the CLI, see *Using the CLI Editor in Configuration Mode* in the CLI User Guide.

To configure Device PE1:

1. Configure the device interfaces including the loopback interface of the device.

```
[edit interfaces]
user@PE1# set ge-0/0/0 unit 0 family inet address 10.0.0.2/8
user@PE1# set ge-0/0/1 unit 0 family inet address 192.0.2.1/24
user@PE1# set ge-0/0/1 unit 0 family mpls
user@PE1# set lo0 unit 0 family inet address 10.127.0.2/8/32
```

2. Configure a static route for routes from Device PE1 with Device P1 as the next-hop destination.

```
[edit routing-options]
user@PE1# set static route 172.16.0.0/16 next-hop 192.0.2.2
```

3. Configure the router ID and autonomous system number for Device PE1.

```
[edit routing-options]
user@PE1# set router-id 10.127.0.2
user@PE1# set autonomous-system 65100
```

4. Configure IBGP peering between the PE devices.

```
[edit protocols]
user@PE1# set bgp group IBGP type internal
user@PE1# set bgp group IBGP local-address 10.127.0.2
user@PE1# set bgp group IBGP family inet-vpn unicast
user@PE1# set bgp group IBGP neighbor 10.127.0.4
```

5. Configure OSPF on all the interfaces of Device PE1, excluding the management interface.

```
[edit protocols]
user@PE1# set ospf area 0.0.0.0 interface ge-0/0/1.0
user@PE1# set ospf area 0.0.0.0 interface lo0.0 passive
```

6. Enable next-hop-based dynamic GRE tunnel configuration on Device PE1.

```
[edit routing-options]
user@PE1# set dynamic-tunnels gre next-hop-based-tunnel
```

7. Configure the dynamic GRE tunnel parameters from Device PE1 to Device PE2.

```
[edit routing-options]
user@PE1# set dynamic-tunnels gre-dyn-tunnel-to-pe2 source-address 10.127.0.2
user@PE1# set dynamic-tunnels gre-dyn-tunnel-to-pe2 gre
user@PE1# set dynamic-tunnels gre-dyn-tunnel-to-pe2 destination-networks 10.127.0.0/16
```

8. Export the load-balancing policy to the forwarding table.

```
[edit routing-options]
user@PE1# set forwarding-table export pplb
```

9. Configure a VRF routing instance on Device PE1 and other routing instance parameters.

```
[edit routing-instances]
user@PE1# set L3VPN-Over-GRE-PE1 instance-type vrf
user@PE1# set L3VPN-Over-GRE-PE1 interface ge-0/0/0.0
user@PE1# set L3VPN-Over-GRE-PE1 route-distinguisher 10.127.0.2:1
user@PE1# set L3VPN-Over-GRE-PE1 vrf-target target:600:1
```

10. Enable BGP in the routing instance configuration for peering with Device CE1.

```
[edit routing-instances]
user@PE1# set L3VPN-Over-GRE-PE1 protocols bgp group pe1-ce1 peer-as 65200
user@PE1# set L3VPN-Over-GRE-PE1 protocols bgp group pe1-ce1 neighbor 10.0.0.1 as-override
```

Results

From configuration mode, confirm your configuration by entering the `show interfaces`, `show routing-options`, `show protocols`, and `show routing-instances` commands. If the output does not display the intended configuration, repeat the instructions in this example to correct the configuration.

```
user@PE1# show interfaces
ge-0/0/0 {
    unit 0 {
        family inet {
            address 10.0.0.2/8;
        }
    }
}
ge-0/0/1 {
    unit 0 {
        family inet {
            address 192.0.2.1/24;
        }
        family mpls;
    }
}
lo0 {
    unit 0 {
        family inet {
            address 10.127.0.2/32;
        }
    }
}
```

```
user@PE1# show routing-options
static {
    route 172.16.0.0/16 next-hop 192.0.2.2;
}
router-id 10.127.0.2;
autonomous-system 65100;
dynamic-tunnels {
    gre next-hop-based-tunnel;
    gre-dyn-tunnel-to-pe2 {
        source-address 10.127.0.2;
        gre;
```

```
        destination-networks {
            10.127.0.0/16;
        }
    }
}
```

```
user@PE1# show protocols
bgp {
    group IBGP {
        type internal;
        local-address 127.0.0.2;
        family inet-vpn {
            unicast;
        }
        neighbor 127.0.0.4;
    }
}
ospf {
    area 0.0.0.0 {
        interface ge-0/0/1.0;
        interface lo0.0 {
            passive;
        }
    }
}
```

```
user@PE1# show routing-instances
L3VPN-Over-GRE-PE1 {
    instance-type vrf;
    interface ge-0/0/0.0;
    route-distinguisher 127.0.0.2:1;
    vrf-target target:600:1;
    protocols {
        bgp {
            group pe1-ce1 {
                peer-as 200;
                neighbor 10.0.0.1 {
                    as-override;
                }
            }
```

```
        }
    }
}
```

If you are done configuring the device, enter `commit` from configuration mode.

Verification

IN THIS SECTION

Confirm that the configuration is working properly.

Verifying the Connection Between PE Devices

Purpose

Verify the BGP peering status between Device PE1 and Device PE2, and the BGP routes received from Device PE2.

Action

From operational mode, run the **show bgp summary** and **show route receive-protocol bgp** ***ip-address*** **table bgp.l3vpn.0** commands.

```
user@PE1> show bgp summary
Groups: 2 Peers: 2 Down peers: 0
Table          Tot Paths  Act Paths Suppressed    History Damp State    Pending
bgp.l3vpn.0
                       2          2          0          0          0          0
Peer                     AS      InPkt     OutPkt    OutQ   Flaps Last Up/Dwn State|#Active/
Received/Accepted/Damped...
127.0.0.4              100        139        136       0       0       58:23 Establ
```

```
  bgp.l3vpn.0: 2/2/2/0
  L3VPN-Over-GRE-PE1.inet.0: 2/2/2/0
10.0.0.1              200        135        136       0       0       58:53 Establ
  L3VPN-Over-GRE-PE1.inet.0: 1/1/1/0
```

```
user@PE1> show route receive-protocol bgp 10.127.0.4 table bgp.l3vpn.0
bgp.l3vpn.0: 2 destinations, 2 routes (2 active, 0 holddown, 0 hidden)
  Prefix                  Nexthop              MED     Lclpref    AS path
  10.127.0.4:1:127.0.0.5/8
*                         127.0.0.4                 65100       65200 I
  10.127.0.4:1:203.0.113.0/24
*                         127.0.0.4                 65100       I
```

Meaning

- In the first output, the BGP session state is `Establ`, which means that the session is up and the PE devices are peered.
- In the second output, Device PE1 has learned two BGP routes from Device PE2.

Verify the Dynamic Tunnel Routes on Device PE1

Purpose

Verify the routes in the inet.3 routing table and the dynamic tunnel database information on Device PE1.

Action

From operational mode, run the **show route table inet.3** and **show dynamic-tunnels database terse** commands.

```
user@PE1> show route table inet.3
inet.3: 2 destinations, 2 routes (2 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

10.127.0.0/8      *[Tunnel/300] 01:01:45
                      Tunnel
```

```
10.127.0.4/8     *[Tunnel/300] 00:10:24
                      Tunnel Composite
```

```
user@PE1> show dynamic-tunnels database terse
Table: inet.3

Destination-network: 10.127.0.0/8
Destination         Source          Next-hop                 Type        Status
10.127.0.4/8      10.127.0.2     0xb395e70 nhid 612        gre         Up
```

Meaning

- In the first output, because Device PE1 is configured with the next-hop-based dynamic GRE tunnel, a tunnel composite route is created for the inet.3 routing table route entry.
- In the second output, the dynamic GRE tunnel created from Device PE1 to Device PE2 is up and has a next-hop ID assigned to it instead of a next-hop interface.

Verify the Dynamic Tunnel Routes on Device PE2

Purpose

Verify the routes in the inet.3 routing table and the dynamic tunnel database information on Device PE2.

Action

From operational mode, run the **show route table inet.3** and **show dynamic-tunnels database terse** commands.

```
user@PE2> show route table inet.3
inet.3: 2 destinations, 2 routes (2 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

10.127.0.0/8      *[Tunnel/300] 01:06:52
                      Tunnel
```

```
10.127.0.2/8     *[Tunnel/300] 01:04:45
                    > via gr-0/1/0.32769
```

```
user@PE1> show dynamic-tunnels database terse
Table: inet.3

Destination-network: 127.0.0.0/8
Destination         Source          Next-hop                  Type        10.
10.127.0.2/8      10.127.0.4     gr-0/1/0.32769            gre         Up
```

Meaning

- In the first output, because Device PE2 has the default interface-based dynamic GRE tunnel configuration, a new interface is created for the inet.3 routing table route entry.
- In the second output, the dynamic GRE tunnel created from Device PE2 to Device PE1 is up, and has the newly created interface name assigned to it.

Verifying That the Routes Have the Expected Indirect-Next-Hop Flag

Purpose

Verify that Device PE1 and Device PE2 are configured to maintain the indirect next hop to forwarding next-hop binding on the Packet Forwarding Engine forwarding table.

Action

From operational mode, run the **show krt indirect-next-hop** command on Device PE1 and Device PE2.

```
user@PE1> show krt indirect-next-hop
Indirect Nexthop:
Index: 1048574 Protocol next-hop address: 10.127.0.4
  RIB Table: bgp.l3vpn.0
  Label: Push 299792
  Policy Version: 1                     References: 2
  Locks: 3                              0xb2ab630
  Flags: 0x0
  INH Session ID: 0x0
  INH Version ID: 0
  Ref RIB Table: unknown
```

```
          Tunnel type: GRE, Reference count: 3, nhid: 612
          Destination address: 10.127.0.4, Source address: 10.127.0.2
          Tunnel id: 1, VPN Label: Push 299792, TTL action: prop-ttl
        IGP FRR Interesting proto count : 2
        Chain IGP FRR Node Num          : 1
           IGP Resolver node(hex)       : 0xb3c6d9c
           IGP Route handle(hex)        : 0xb1ad230      IGP rt_entry protocol        : Tunnel
           IGP Actual Route handle(hex) : 0x0            IGP Actual rt_entry protocol : Any
```

```
user@PE2> show krt indirect-next-hop
Indirect Nexthop:
Index: 1048574 Protocol next-hop address: 10.127.0.2
  RIB Table: bgp.l3vpn.0
  Label: Push 299792
  Policy Version: 2                      References: 2
  Locks: 3                               0xb2ab630
  Flags: 0x2
  INH Session ID: 0x145
  INH Version ID: 0
  Ref RIB Table: unknown
        Next hop: via gr-0/1/0.32769
        Label operation: Push 299792
        Label TTL action: prop-ttl
        Load balance label: Label 299792: None;
        Label element ptr: 0xb395d40
        Label parent element ptr: 0x0
        Label element references: 1
        Label element child references: 0
        Label element lsp id: 0
        Session Id: 0x144
      IGP FRR Interesting proto count : 2
      Chain IGP FRR Node Num          : 1
         IGP Resolver node(hex)       : 0xb3d36e8
         IGP Route handle(hex)        : 0xb1af060      IGP rt_entry protocol        : Tunnel
         IGP Actual Route handle(hex) : 0x0            IGP Actual rt_entry protocol : Any
```

Meaning

- In the first output, Device PE1 has a next-hop-based dynamic GRE tunnel to Device PE2.
- In the second output, Device PE2 has an interface-based dynamic GRE tunnel to device PE1.

Troubleshooting

IN THIS SECTION

To troubleshoot the next-hop-based dynamic tunnels, see:

Troubleshooting Commands

Problem

The next-hop-based dynamic GRE tunnel configuration is not taking effect.

Solution

To troubleshoot the next-hop-based GRE tunnel configuration, use the following `traceoptions` commands at the `[edit routing-options dynamic-tunnels]` statement hierarchy:

- `traceoptions file` *`file-name`*
- `traceoptions file size` *`file-size`*
- `traceoptions flag` *`all`*

For example:

```
[edit routing-options dynamic-tunnels]
traceoptions {
    file gre_dyn_pe1.wri size 4294967295;
    flag all;
}
```

SEE ALSO

Configuring GRE Tunnels for Layer 3 VPNs | 880

Example: Configuring Next-Hop-Based MPLS-Over-UDP Dynamic Tunnels

IN THIS SECTION

This example shows how to configure a dynamic MPLS-over-UDP tunnel that includes a tunnel composite next hop. The MPLS-over-UDP feature provides a scaling advantage on the number of IP tunnels supported on a device.

Starting in Junos OS Release 18.3R1, MPLS-over-UDP tunnels are supported on PTX Series routers and QFX Series switches. For every dynamic tunnel configured on a PTX router or a QFX switch, a tunnel composite next hop, an indirect next hop, and a forwarding next hop is created to resolve the tunnel destination route. You can also use policy control to resolve the dynamic tunnel over select prefixes by including the forwarding-rib configuration statement at the `[edit routing-options dynamic-tunnels]` hierarchy level.

Requirements

This example uses the following hardware and software components:

- Five MX Series routers with MPCs and MICs.
- Junos OS Release 16.2 or later running on the provider edge (PE) routers.

Before you begin:

1. Configure the device interfaces, including the loopback interface.
2. Configure the router ID and autonmous system number for the device.
3. Establish an internal BGP (IBGP) session with the remote PE device.
4. Establish OSPF peering among the devices.

Overview

IN THIS SECTION

Starting with Junos OS Release 16.2, a dynamic UDP tunnel supports the creation of a tunnel composite next hop for every UDP tunnel configured. These next-hop-based dynamic UDP tunnels are referred to as MPLS-over-UDP tunnels. The tunnel composite next hop are enabled by default for the MPLS-over-UDP tunnels.

MPLS-over-UDP tunnels can be bidirectional or unidirectional in nature.

- Bidirectional—When the PE devices are connected over MPLS-over-UDP tunnels in both directions, it is called a bidirectional MPLS-over-UDP tunnel.

- Unidirectional—When two PE devices are connected over MPLS-over-UDP tunnel in one direction, and over MPLS/IGP in the other direction, it is called an unidirectional MPLS-over-UDP tunnel.

  Unidirectional MPLS-over-UDP tunnels are used in migration scenarios, or in cases where two PE devices provide connectivity to each other over two disjoint networks. Because reverse direction tunnel does not exist for unidirectional MPLS-over-UDP tunnels, you must configure a filter-based MPLS-over-UDP decapsulation on the remote PE device for forwarding the traffic.

Starting in Junos OS Release 18.2R1, on PTX series routers and QFX10000 with unidirectional MPLS-over-UDP tunnels, you must configure the remote PE device with an input filter for MPLS-over-UDP packets, and an action for decapsulating the IP and UDP headers for forwarding the packets in the reverse tunnel direction.

For example, on the remote PE device, Device PE2, the following configuration is required for unidirectional MPLS-over-UDP tunnels:

PE2

```
[edit firewall filter]
user@host# set Decap_Filter term udp_decap from protocol udp
user@host# set Decap_Filter term udp_decap from destination-port 6635
user@host# set Decap_Filter term udp_decap then count UDP_PKTS
user@host# set Decap_Filter term udp_decap then decapsulate mpls-in-udp
user@host# set Decap_Filter term def then count def_pkt
user@host# set Decap_Filter term def then accept
```

In the above sample configuration, *Decap_Filter* is the name of the firewall filter used for MPLS-over-UDP decapsulation. The term *udp_decap* is the input filter for accepting UDP packets on the core-facing interface of Device PE2, and then decapsulate the MPLS-over-UDP packets to MPLS-over-IP packets for forwarding.

You can use the existing firewall operational mode commands, such as `show firewall filter` to view the filter-based MPLS-over-UDP decapsulation.

For example:

```
user@host >show firewall filter Decap_Filter
Filter: Decap_Filter
Counters:
Name                                                Bytes              Packets
UDP_PKTS                                            16744                  149
def_pkt                                             13049                  136
```

NOTE: For unidirectional MPLS-over-UDP tunnels:

- Only IPv4 address is supported as the outer header. Filter-based MPLS-over-UDP decapsulation does not support IPv6 address in the outer header.
- Only the default routing instance is supported after decapsulation.

Starting in Junos OS Release 17.1, on MX Series routers with MPCs and MICs, the scaling limit of MPLS-over-UDP tunnels is increased.

Starting in Junos Release 19.2R1, on MX Series routers with MPCs and MICs, carrier supporting carrier (CSC) architecture can be deployed with MPLS-over-UDP tunnels carrying MPLS traffic over dynamic IPv4 UDP tunnels that are established between supporting carrier's PE devices. With this enhancement, the scaling advantage that the MPLS-over-UDP tunnels provided is further increased. The CSC support with MPLS-over-UDP tunnel is not supported for IPv6 UDP tunnel.

The existing dynamic tunnel feature requires complete static configuration. Currently, the tunnel information received from peer devices in advertised routes is ignored. Starting in Junos OS Release 17.4R1, on MX Series routers, the next-hop-based dynamic MPLS-over-UDP tunnels are signaled using BGP encapsulation extended community. BGP export policy is used to specify the tunnel types, advertise the sender side tunnel information, and parse and convey the receiver side tunnel information. A tunnel is created according to the received type tunnel community.

Multiple tunnel encapsulations are supported by BGP. On receiving multiple capability, the next-hop-based dynamic tunnel is created based on the configured BGP policy and tunnel preference. The tunnel preference should be consistent across both the tunnel ends for the tunnel to be set up. By default, MPLS-over-UDP tunnel is preferred over GRE tunnels. If dynamic tunnel configuration exists, it takes precedence over received tunnel community.

When configuring a next-hop-based dynamic MPLS-over-UDP tunnel, be aware of the following considerations:

- An IBGP session must be configured between the PE devices.
- A switchover between the next-hop-based dynamic tunnel encapsulations (UDP and GRE) is allowed, and this can impact network performance in terms of the supported IP tunnel scaling values in each mode.
- Having both GRE and UDP next-hop-based dynamic tunnel encapsulation types for the same tunnel destination leads to a commit failure.
- For unidirectional MPLS-over-UDP tunnels, you must explicitly configure filter-based MPLS-over-UDP decapsulation on the remote PE device for the packets to be forwarded.
- Graceful Routing Engine switchover (GRES) is supported with MPLS-over-UDP, and the MPLS-over-UDP tunnel type flags are unified ISSU and NSR compliant.
- MPLS-over-UDP tunnels are supported on virtual MX (vMX) in Lite mode.
- MPLS-over-UDP tunnels support dynamic GRE tunnel creation based upon new IPv4-mapped-IPv6 next hops.
- MPLS-over-UDP tunnel are supported in interoperability with contrail, wherein the MPLS-over-UDP tunnels are created from the contrail vRouter to an MX gateway. To enable this, the following community is required to be advertised in the route from the MX Series router to the contrail vRouter:

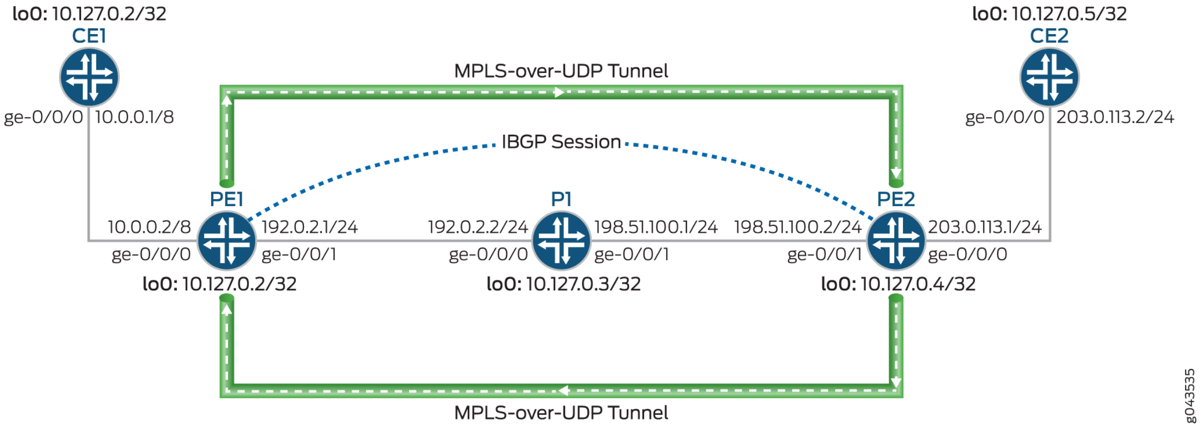
  ```
  [edit policy-options community]
  udp members 0x030c:64512:13;
  ```

  At a given point in time, only one tunnel type is supported on the contrail vRouter—next-hop-based dynamic GRE tunnels, MPLS-over-UDP tunnels, or VXLAN.
- The following features are not supported with the next-hop-based dynamic MPLS-over-UDP tunnel configuration:
  - RSVP automatic mesh
  - Plain IPV6 GRE and UDP tunnel configuration
  - Logical systems

Topology

Figure 74 on page 925 illustrates a Layer 3 VPN scenario over dynamic MPLS-over-UDP tunnels. The customer edge (CE) devices, CE1 and CE2, connect to provider edge (PE) devices, PE1 and PE2, respectively. The PE devices are connected to a provider device (Device P1), and an internal BGP (IBGP) session interconnects the two PE devices. A dynamic next-hop-based bidirectional MPL-over-UDP tunnel is configured between the PE devices.

Figure 74: Dynamic MPLS-over-UDP Tunnels



The MPLS-over-UDP tunnel is handled as follows:

1. After a MPLS-over-UDP tunnel is configured, a tunnel destination mask route with a tunnel composite next hop is created for the tunnel in the inet.3 routing table. This IP tunnel route is withdrawn only when the dynamic tunnel configuration is deleted.

   The tunnel composite next-hop attributes include the following:

   - When Layer 3 VPN composite next hop is disabled—Source and destination address, encapsulation string, and VPN label.

   - When Layer 3 VPN composite next hop and per-prefix VPN label allocation are enabled—Source address, destination address, and encapsulation string.

   - When Layer 3 VPN composite next hop is enabled and per-prefix VPN label allocation is disabled —Source address, destination address, and encapsulation string. The route in this case is added to the other virtual routing and forwarding instance table with a secondary route.

2. The PE devices are interconnected using an IBGP session. The IBGP route next hop to a remote BGP neighbor is the protocol next hop, which is resolved using the tunnel mask route with the tunnel next hop.

3. After the protocol next hop is resolved over the tunnel composite next hop, indirect next hops with forwarding next hops are created.

4. The tunnel composite next hop is used to forward the next hops of the indirect next hops.

Configuration

IN THIS SECTION

CLI Quick Configuration

To quickly configure this example, copy the following commands, paste them into a text file, remove any line breaks, change any details necessary to match your network configuration, copy and paste the commands into the CLI at the `[edit]` hierarchy level, and then enter `commit` from configuration mode.

CE1

```
set interfaces ge-0/0/0 unit 0 family inet address 10.0.0.1/8
set interfaces lo0 unit 0 family inet address 10.127.0.1/32
set routing-options router-id 10.127.0.1
set routing-options autonomous-system 65200
set protocols bgp group ce1-pe1 export export-loopback-direct
set protocols bgp group ce1-pe1 peer-as 100
set protocols bgp group ce1-pe1 neighbor 10.0.0.2
set policy-options policy-statement export-loopback-direct term term-1 from interface lo0.0
set policy-options policy-statement export-loopback-direct term term-1 from route-filter
10.127.0.1/32 exact
set policy-options policy-statement export-loopback-direct term term-1 then accept
```

CE2

```
set interfaces ge-0/0/0 unit 0 family inet address 203.0.113.2/24
set interfaces lo0 unit 0 family inet address 10.127.0.5/32
set routing-options router-id 10.127.0.5
```

```
set routing-options autonomous-system 65200
set protocols bgp group ce1-pe1 export export-loopback-direct
set protocols bgp group ce1-pe1 peer-as 65100
set protocols bgp group ce1-pe1 neighbor 203.0.113.1
set policy-options policy-statement export-loopback-direct term term-1 from interface lo0.0
set policy-options policy-statement export-loopback-direct term term-1 from route-filter
10.127.0.5/32 exact
set policy-options policy-statement export-loopback-direct term term-1 then accept
```

PE1

```
set interfaces ge-0/0/0 unit 0 family inet address 10.0.0.2/8
set interfaces ge-0/0/1 unit 0 family inet address 192.0.2.1/24
set interfaces ge-0/0/1 unit 0 family mpls
set interfaces lo0 unit 0 family inet address 10.127.0.2/32
set routing-options static route 10.33.0/16 next-hop 192.0.2.2
set routing-options router-id 10.127.0.2
set routing-options autonomous-system 65100
set routing-options forwarding-table export pplb
set routing-options dynamic-tunnels gre next-hop-based-tunnel
set routing-options dynamic-tunnels udp-dyn-tunnel-to-pe2 source-address 10.127.0.2
set routing-options dynamic-tunnels udp-dyn-tunnel-to-pe2 udp
set routing-options dynamic-tunnels udp-dyn-tunnel-to-pe2 destination-networks 10.127.0.0/24
set protocols bgp group IBGP type internal
set protocols bgp group IBGP local-address 10.127.0.2
set protocols bgp group IBGP family inet-vpn unicast
set protocols bgp group IBGP neighbor 10.127.0.4
set protocols ospf area 0.0.0.0 interface ge-0/0/1.0
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set routing-instances MPLS-over-UDP-PE1 instance-type vrf
set routing-instances MPLS-over-UDP-PE1 interface ge-0/0/0.0
set routing-instances MPLS-over-UDP-PE1 route-distinguisher 10.127.0.2:1
set routing-instances MPLS-over-UDP-PE1 vrf-target target:600:1
set routing-instances MPLS-over-UDP-PE1 protocols bgp group pe1-ce1 peer-as 65200
set routing-instances MPLS-over-UDP-PE1 protocols bgp group pe1-ce1 neighbor 10.0.0.1 as-override
```

P1

```
set interfaces ge-0/0/0 unit 0 family inet address 192.0.2.2/24
set interfaces ge-0/0/0 unit 0 family mpls
set interfaces ge-0/0/1 unit 0 family inet address 198.51.100.1/24
```

```
set interfaces ge-0/0/1 unit 0 family mpls
set interfaces lo0 unit 0 family inet address 10.127.0.3/32
set routing-options router-id 10.127.0.3
set routing-options autonomous-system 65100
set protocols ospf area 0.0.0.0 interface ge-0/0/0.0
set protocols ospf area 0.0.0.0 interface ge-0/0/1.0
set protocols ospf area 0.0.0.0 interface lo0.0 passive
```

PE2

```
set interfaces ge-0/0/0 unit 0 family inet address 203.0.113.1/24
set interfaces ge-0/0/1 unit 0 family inet address 198.51.100.2/24
set interfaces ge-0/0/1 unit 0 family mpls
set interfaces lo0 unit 0 family inet address 10.127.0.4/8
set routing-options nonstop-routing
set routing-options router-id 10.127.0.4
set routing-options autonomous-system 65100
set routing-options forwarding-table export pplb
set routing-options dynamic-tunnels udp-dyn-tunnel-to-pe1 source-address 10.127.0.4
set routing-options dynamic-tunnels udp-dyn-tunnel-to-pe1 udp
set routing-options dynamic-tunnels udp-dyn-tunnel-to-pe1 destination-networks 10.127.0.0/24
set protocols bgp group IBGP type internal
set protocols bgp group IBGP local-address 10.127.0.4
set protocols bgp group IBGP family inet-vpn unicast
set protocols bgp group IBGP neighbor 10.127.0.2
set protocols ospf area 0.0.0.0 interface ge-0/0/1.0
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set routing-instances MPLS-over-UDP-PE2 instance-type vrf
set routing-instances MPLS-over-UDP-PE2 interface ge-0/0/0.0
set routing-instances MPLS-over-UDP-PE2 route-distinguisher 10.127.0.4:1
set routing-instances MPLS-over-UDP-PE2 vrf-target target:600:1
set routing-instances MPLS-over-UDP-PE2 protocols bgp group ebgp peer-as 65200
set routing-instances MPLS-over-UDP-PE2 protocols bgp group ebgp neighbor 203.0.113.2 as-override
```

Procedure

Step-by-Step Procedure

The following example requires that you navigate various levels in the configuration hierarchy. For information about navigating the CLI, see *Using the CLI Editor in Configuration Mode* in the CLI User Guide.

To configure Device PE1:

1. Configure the device interfaces including the loopback interface of the device.

   ```
   [edit interfaces]
   user@PE1# set ge-0/0/0 unit 0 family inet address 10.0.0.2/8
   user@PE1# set ge-0/0/1 unit 0 family inet address 192.0.2.1/24
   user@PE1# set ge-0/0/1 unit 0 family mpls
   user@PE1# set lo0 unit 0 family inet address 10.127.0.2/8
   ```

2. Configure a static route for routes from Device PE1 with Device P1 as the next-hop destination.

   ```
   [edit routing-options]
   user@PE1# set static route 10.33.0.0/16 next-hop 192.0.2.2
   ```

3. Configure the router-ID and autonomous system number for Device PE1.

   ```
   [edit routing-options]
   user@PE1# set router-id 10.127.0.2
   user@PE1# set autonomous-system 65100
   ```

4. (PTX Series only) Configure policy control to resolve the MPLS-over-UDP dynamic tunnel route over select prefixes.

   ```
   [edit routing-options dynamic-tunnels]
   user@PTX-PE1# set forwarding-rib inet.0 inet-import dynamic-tunnel-fwd-route-import
   ```

5. (PTX Series only) Configure the inet-import policy for resolving dynamic tunnel destination routes over .

   ```
   [edit policy-options]
   user@PTX-PE1# set policy-statement dynamic-tunnel-fwd-route-import term 1 from route-filter
   10.127.0.4/32 exact
   user@PTX-PE1# set policy-statement dynamic-tunnel-fwd-route-import term 1 then accept
   user@PTX-PE1# set policy-options policy-statement dynamic-tunnel-fwd-route-import then
   reject
   ```

6. Configure IBGP peering between the PE devices.

```
[edit protocols]
user@PE1# set bgp group IBGP type internal
user@PE1# set bgp group IBGP local-address 10.127.0.2
user@PE1# set bgp group IBGP family inet-vpn unicast
user@PE1# set bgp group IBGP neighbor 10.127.0.4
```

7. Configure OSPF on all the interfaces of Device PE1, excluding the management interface.

```
[edit protocols]
user@PE1# set ospf area 0.0.0.0 interface ge-0/0/1.0
user@PE1# set ospf area 0.0.0.0 interface lo0.0 passive
```

8. Enable next-hop-based dynamic GRE tunnel configuration on Device PE1.

NOTE: This step is required only for illustrating the implementation difference between next-hop-based dynamic GRE tunnels and MPLS-over-UDP tunnels.

```
[edit routing-options]
user@PE1# set dynamic-tunnels gre next-hop-based-tunnel
```

9. Configure the MPLS-over-UDP tunnel parameters from Device PE1 to Device PE2.

```
[edit routing-options]
user@PE1# set dynamic-tunnels udp-dyn-tunnel-to-pe2 source-address 10.127.0.2
user@PE1# set dynamic-tunnels udp-dyn-tunnel-to-pe2 udp
user@PE1# set dynamic-tunnels udp-dyn-tunnel-to-pe2 destination-networks 10.127.0.0/24
```

10. Configure a VRF routing instance on Device PE1 and other routing instance parameters.

```
[edit routing-instances]
user@PE1# set MPLS-over-UDP-PE1 instance-type vrf
user@PE1# set MPLS-over-UDP-PE1 interface ge-0/0/0.0
user@PE1# set MPLS-over-UDP-PE1 route-distinguisher 10.127.0.2:1
user@PE1# set MPLS-over-UDP-PE1 vrf-target target:600:1
```

11. Enable BGP in the routing instance configuration for peering with Device CE1.

    ```
    [edit routing-instances]
    user@PE1# set MPLS-over-UDP-PE1 protocols bgp group pe1-ce1 peer-as 65200
    user@PE1# set MPLS-over-UDP-PE1 protocols bgp group pe1-ce1 neighbor 10.0.0.1 as-override
    ```

Results

From configuration mode, confirm your configuration by entering the `show interfaces`, `show routing-options`, `show protocols`, and `show routing-instances` commands. If the output does not display the intended configuration, repeat the instructions in this example to correct the configuration.

```
user@PE1# show interfaces
ge-0/0/0 {
    unit 0 {
        family inet {
            address 10.0.0.2/8;
        }
    }
}
ge-0/0/1 {
    unit 0 {
        family inet {
            address 192.0.2.1/24;
        }
        family mpls;
    }
}
lo0 {
    unit 0 {
        family inet {
            address 10.127.0.2/32;
        }
    }
}
```

```
user@PE1# show routing-options
static {
    route 10.33.0.0/16 next-hop 192.0.2.2;
```

```
}
router-id 10.127.0.2;
autonomous-system 65100;
forwarding-table {
    export pplb;
}
dynamic-tunnels {
    gre next-hop-based-tunnel;
    udp-dyn-tunnel-to-pe2 {
        source-address 10.127.0.2;
        udp;
        destination-networks {
            10.127.0.0/24;
        }
    }
}
```

```
user@PE1# show protocols
bgp {
    group IBGP {
        type internal;
        local-address 10.127.0.2;
        family inet-vpn {
            unicast;
        }
        neighbor 10.127.0.4;
    }
}
ospf {
    area 0.0.0.0 {
        interface ge-0/0/1.0;
        interface lo0.0 {
            passive;
        }
    }
}
```

```
user@PE1# show routing-instances
MPLS-over-UDP-PE1 {
    instance-type vrf;
```

```
    interface ge-0/0/0.0;
    route-distinguisher 10.127.0.2:1;
    vrf-target target:600:1;
    protocols {
        bgp {
            group pe1-ce1 {
                peer-as 65200;
                neighbor 10.0.0.1 {
                    as-override;
                }
            }
        }
    }
}
```

If you are done configuring the device, enter `commit` from configuration mode.

Verification

IN THIS SECTION

Confirm that the configuration is working properly.

Verifying the Connection Between PE Devices

Purpose

Verify the BGP peering status between Device PE1 and Device PE2, and the BGP routes received from Device PE2.

Action

From operational mode, run the **show bgp summary** and **show route receive-protocol bgp *ip-address* table bgp.l3vpn.0** commands.

```
user@PE1> show bgp summary
Groups: 2 Peers: 2 Down peers: 0
Table          Tot Paths  Act Paths Suppressed    History Damp State    Pending
bgp.l3vpn.0
                       2          2          0          0          0          0
Peer                     AS      InPkt     OutPkt    OutQ   Flaps Last Up/Dwn State|#Active/
Received/Accepted/Damped...
10.127.0.4            65100        139        136       0       0       58:23 Establ
  bgp.l3vpn.0: 2/2/2/0
  MPLS-over-UDP-PE1.inet.0: 2/2/2/0
10.10.0.1               65200        135        136       0       0       58:53 Establ
  MPLS-over-UDP-PE1.inet.0: 1/1/1/0
```

```
user@PE1> show route receive-protocol bgp 10.127.0.4 table bgp.l3vpn.0
bgp.l3vpn.0: 2 destinations, 2 routes (2 active, 0 holddown, 0 hidden)
  Prefix                  Nexthop              MED     Lclpref    AS path
  10.127.0.4:1:127.0.0.5/8
*                         10.127.0.4                  65100        65200 I
```

Meaning

- In the first output, the BGP session state is `Establ`, which means that the session is up and the PE devices are peered.
- In the second output, Device PE1 has learned a BGP route from Device PE2.

Verify the Dynamic Tunnel Routes on Device PE1

Purpose

Verify the routes in the inet.3 routing table and the dynamic tunnel database information on Device PE1.

Action

From operational mode, run the **show route table inet.3**, **show dynamic-tunnels database terse**, **show dynamic-tunnels database**, and **show dynamic-tunnels database summary** commands.

```
user@PE1> show route table inet.3
inet.3: 2 destinations, 2 routes (2 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both
10.127.0.0/24     *[Tunnel/300] 00:21:18
                      Tunnel
127.0.0.4/8     *[Tunnel/300] 00:21:18
                      Tunnel Composite
```

```
user@PE1> show dynamic-tunnels database terse
Table: inet.3

Destination-network: 10.127.0.0/24
Destination         Source          Next-hop                  Type        Status
10.127.0.4/8      10.127.0.2     0xb395b10 nhid 613        udp         Up
```

```
user@PE1> show dynamic-tunnels database
Table: inet.3
. . .
Tunnel to: 10.127.0.4/32
  Reference count: 2
  Next-hop type: UDP
    Source address: 10.127.0.2 Tunnel Id: 2
    Next hop: tunnel-composite, 0xb395b10, nhid 613
      VPN Label: Push 299776 Reference count: 3
      Traffic Statistics: Packets 0, Bytes 0
      State: Up
```

```
user@PE1> show dynamic-tunnels database summary
Dynamic Tunnels,  Total 1 displayed
GRE Tunnel:
Active Tunnel Mode, Next Hop Base
  IFL Based,  Total 0 displayed, Up 0, Down 0
  Nexthop Based,  Total 0 displayed, Up 0, Down 0
```

```
RSVP Tunnel:
  Total 0 displayed
UDP Tunnel:
  Total 1 displayed, Up 1, Down 0
```

Meaning

- In the first output, because Device PE1 is configured with the MPLS-over-UDP tunnel, a tunnel composite route is created for the inet.3 routing table route entry.
- In the remaining outputs, the MPLS-over-UDP tunnel is displayed with the tunnel encapsulation type, tunnel next hop parameters, and tunnel status.

Verify the Dynamic Tunnel Routes on Device PE2

Purpose

Verify the routes in the inet.3 routing table and the dynamic tunnel database information on Device PE2.

Action

From operational mode, run the **show route table inet.3**, and the **show dynamic-tunnels database terse** commands.

```
user@PE2> show route table inet.3
inet.3: 2 destinations, 2 routes (2 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

10.127.0.0/24      *[Tunnel/300] 00:39:31
                      Tunnel
10.127.0.2/32     *[Tunnel/300] 00:24:53
                      Tunnel Composite
```

```
user@PE1> show dynamic-tunnels database terse
Table: inet.3

Destination-network: 127.0.0.0/8
```

```
Destination          Source          Next-hop                  Type        Status
10.127.0.2/32      10.127.0.4      0xb395450 nhid 615        udp         Up
```

Meaning

The outputs show the MPLS-over-UDP tunnel creation and the next-hop ID assigned as the next-hop interface, similar to Device PE1.

Verifying That the Routes Have the Expected Indirect-Next-Hop Flag

Purpose

Verify that Device PE1 and Device PE2 are configured to maintain the indirect next hop to forwarding next-hop binding on the Packet Forwarding Engine forwarding table.

Action

From operational mode, run the **show krt indirect-next-hop** command on Device PE1 and Device PE2.

```
user@PE1> show krt indirect-next-hop
Indirect Nexthop:
Index: 1048574 Protocol next-hop address: 10.127.0.4
  RIB Table: bgp.l3vpn.0
  Label: Push 299776
  Policy Version: 1                     References: 1
  Locks: 3                              0xb2ab630
  Flags: 0x0
  INH Session ID: 0x0
  INH Version ID: 0
  Ref RIB Table: unknown
        Tunnel type: UDP, Reference count: 3, nhid: 613
        Destination address: 10.127.0.4, Source address: 10.127.0.2
        Tunnel id: 2, VPN Label: Push 299776, TTL action: prop-ttl
      IGP FRR Interesting proto count : 1
      Chain IGP FRR Node Num          : 1
         IGP Resolver node(hex)       : 0xb3c70dc
```

```
          IGP Route handle(hex)        : 0xb1ae688      IGP rt_entry protocol        : Tunnel
          IGP Actual Route handle(hex) : 0x0            IGP Actual rt_entry protocol : Any
```

```
user@PE2> show krt indirect-next-hop
Indirect Nexthop:
Index: 1048575 Protocol next-hop address: 10.127.0.2
  RIB Table: bgp.l3vpn.0
  Label: Push 299776
  Policy Version: 1                     References: 2
  Locks: 3                              0xb2ab740
  Flags: 0x0
  INH Session ID: 0x0
  INH Version ID: 0
  Ref RIB Table: unknown
        Tunnel type: UDP, Reference count: 3, nhid: 615
        Destination address: 10.127.0.2, Source address: 10.127.0.4
        Tunnel id: 1, VPN Label: Push 299776, TTL action: prop-ttl
      IGP FRR Interesting proto count : 2
      Chain IGP FRR Node Num          : 1
         IGP Resolver node(hex)       : 0xb3d3a28
         IGP Route handle(hex)        : 0xb1ae634      IGP rt_entry protocol        : Tunnel
         IGP Actual Route handle(hex) : 0x0            IGP Actual rt_entry protocol : Any
```

Meaning

The outputs show that a next-hop-based dynamic MPLS-over-UDP tunnel is created between the PE devices.

Troubleshooting

IN THIS SECTION

To troubleshoot the next-hop-based dynamic tunnels, see:

Troubleshooting Commands

Problem

The next-hop-based dynamic MPLS-over-UDP tunnel configuration is not taking effect.

Solution

To troubleshoot the next-hop-based MPLS-over-UDP tunnel configuration, use the following `traceroute` commands at the `[edit routing-options dynamic-tunnels]` statement hierarchy:

- `traceoptions file` *`file-name`*
- `traceoptions file size` *`file-size`*
- `traceoptions flag all`

For example:

```
[edit routing-options dynamic-tunnels]
traceoptions {
    file udp_dyn_pe1.wri size 4294967295;
    flag all;
}
```

Anti-Spoofing Protection for Next-Hop-Based Dynamic Tunnels Overview

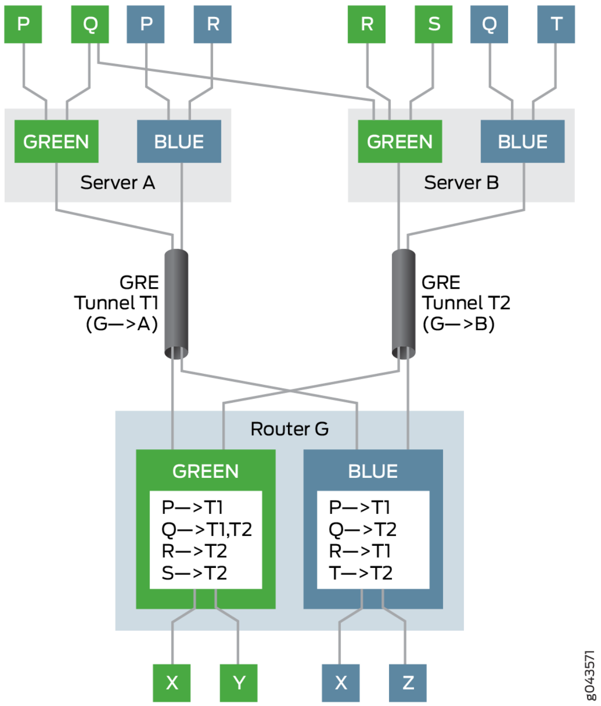
With the rise in deployment of high-scale IP tunnels in data centers, there is a need to add security measures that allow users to limit malicious traffic from compromised virtual machines (VMs). One possible attack is the injecting of traffic into an arbitrary customer VPN from a compromised server through the gateway router. In such cases, anti-spoofing checks on IP tunnels ensure that only legitimate sources are injecting traffic into data centers from their designated IP tunnels.

Next-hop-based dynamic IP tunnels create a tunnel composite next hop for every dynamic tunnel created on the device. Because next-hop-based dynamic tunnels remove the dependency on physical interfaces for every new dynamic tunnel configured, configuring next-hop-based dynamic tunnels provides a scaling advantage over the number of dynamic tunnels that can be created on a device. Starting in Junos OS Release 17.1, anti-spoofing capabilities for next-hop-based dynamic IP tunnels is provided for next-hop-based dynamic tunnels. With this enhancement, a security measure is implemented to prevent injecting of traffic into an arbitrary customer VPN from a compromised server through the gateway router.

Anti-spoofing is implemented using reverse path forwarding checks in the Packet Forwarding Engine. The checks are implemented for the traffic coming through the tunnel to the routing instance. Currently, when the gateway router receives traffic from a tunnel, only the destination lookup us done and the packet is forwarded accordingly. When anti-spoofing protection is enabled, the gateway router also does a source address lookup of the encapsulation packet IP header in the VPN, in addition to the tunnel destination lookup. This ensures that legitimate sources are injecting traffic through their designated IP tunnels. As a result, anti-spoofing protection ensures that the tunnel traffic is received from a legitimate source on the designated tunnels.

Figure 75 on page 940 illustrates a sample topology with the requirements for anti-spoofing protection.

Figure 75: Anti-Spoofing Protection for Next-Hop-Based Dynamic Tunnels



In this example, the gateway router is Router G. Router G has two VPNs—Green and Blue. The two servers, Server A and Server B, can reach the Green and Blue VPNs on Router G through the next-hop-based dynamic tunnels T1 and T2, respectively. Several hosts and virtual machines (P, Q, R, S, and T) connected to the servers can reach the VPNs through the gateway router, Router G. Router G has the virtual routing and forwarding (VRF) tables for Green and Blue VPNs, each populated with the reachability information for the virtual machines in those VPNs.

For example, in VPN Green, Router G uses tunnel T1 to reach host P, tunnel T2 to reach hosts R and S, and load balancing is done between tunnels T1 and T2 to reach the multihomed host Q. In VPN Blue, Router G uses tunnel T1 to reach hosts P and R, and tunnel T2 to reach hosts Q and T.

The check passes for reverse path forwarding when:

- A packet comes from a legitimate source on its designated tunnel.

  Host P in VPN Green sends a packet to host X using tunnel T1. Because Router G can reach host P through tunnel T1, it allows the packet to pass and forwards the packet to host X.

- A packet comes from a multihomed source on its designated tunnels.

  Host Q in VPN Green is multihomed on servers A and B, and can reach Router G through tunnels T1 and T2. Host Q sends a packet to host Y using tunnel T1, and a packet to host X using tunnel T2. Because Router G can reach host Q through tunnels T1 and T2, it allows the packets to pass and forwards them to hosts Y and X, respectively.

Layer 3 VPNs do not have anti-spoofing protection enabled by default. To enable anti-spoofing for next-hop-based dynamic tunnels, include the `ip-tunnel-rpf-check` statement at the `[edit routing-instances` *`routing-instance-name`* `routing-options forwarding-table]` hierarchy level. The reverse path forwarding check is applied to the VRF routing instance only. The default mode is set to `strict`, where the packet that comes from a source on a nondesignated tunnel does not pass the check. The `ip-tunnel-rpf-check` mode can be set as `loose`, where the reverse path forwarding check fails when the packet comes from a nonexistent source. An optional firewall filter can be configured under the `ip-tunnel-rpf-check` statement to count and log the packets that failed the reverse path forwarding check.

The following sample output shows an anti-spoofing configuration:

```
[edit routing-instances routing-instance-name routing-options forwarding-table]
ip-tunnel-rpf-check {
    mode loose;
    fail-filter filter-name;
}
```

Take the following guidelines under consideration when configuring anti-spoofing protection for next-hop-based dynamic tunnels:

- Anti-spoofing protection can be enabled for IPv4 tunnels and IPv4 data traffic only. The anti-spoofing capabilities are not supported on IPv6 tunnels and IPv6 data traffic.

- Anti-spoofing for next-hop-based dynamic tunnels can detect and prevent a compromised virtual machine (inner source reverse path forwarding check) but not a compromised server that is label-spoofing.

- The next-hop-based IP tunnels can originate and terminate on an inet.0 routing table.

- Anti-spoofing protection is effective when the VRF routing instance has label-switched interfaces (LSIs) (using the `vrf-table-label`), or virtual tunnel (VT) interfaces. With `per-next-hop` label on the VRF routing instance, anti-spoofing protection is not supported.

- The `rpf fail-filter` is applicable only to the inner IP packet.
- Enabling anti-spoofing checks does not affect the scaling limit of the next-hop-based dynamic tunnels on a device.
- The system resource utilization with anti-spoofing protection enabled for the VRF routing instance is slightly higher than the utilization of next-hop-based dynamic tunnels without the anti-spoofing protection enabled.
- Anti-spoofing protection requires additional source IP address checks, which has minimal impact on network performance.
- Graceful Routing Engine switchover (GRES) and in-service software upgrade (ISSU) are supported with anti-spoofing protection.

Example: Configuring Anti-Spoofing Protection for Next-Hop-Based Dynamic Tunnels

IN THIS SECTION

This example shows how to configure reverse path forwarding checks for the virtual routing and forwarding (VRF) routing instance to enable anti-spoofing protection for next-hop-based dynamic tunnels. The checks ensure that legitimate sources are injecting traffic through their designated IP tunnels.

Requirements

This example uses the following hardware and software components:

- Three MX Series Routers with MICs, each connected to a host device.
- Junos OS Release 17.1 or later running on one or all the routers.

Before you begin:

- Enable tunnel services configuration on the Flexible PIC Concentrator.
- Configure the router interfaces.

- Configure the router-ID and assign an autonomous system number for the router.
- Establish an internal BGP (IBGP) session with the tunnel endpoints.
- Configure RSVP on all the routers.
- Configure OSPF or any other interior gateway protocol on all the routers.
- Configure two dynamic next-hop-based IP tunnels between the two routers.
- Configure a VRF routing instance for every router-to-host connection.

Overview

IN THIS SECTION

Starting in Junos OS Release 17.1, anti-spoofing capabilities are added to next-hop-based dynamic IP tunnels, where checks are implemented for the traffic coming through the tunnel to the routing instance using reverse path forwarding in the Packet Forwarding Engine.

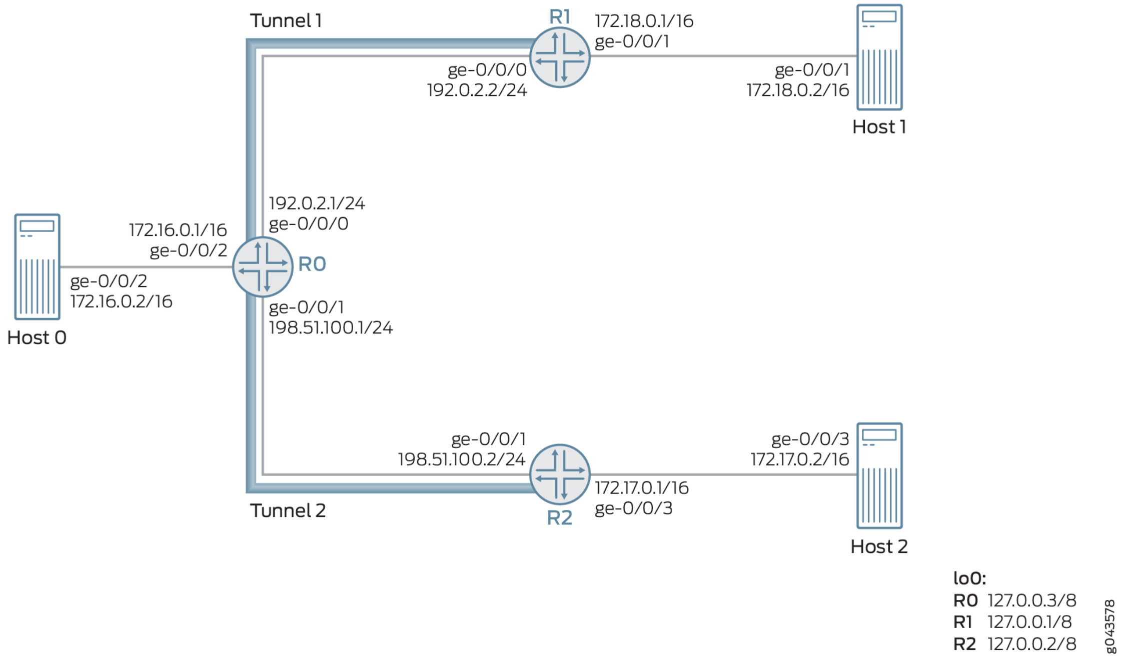
Currently, when the gateway router receives traffic from a tunnel, only the destination address lookup is done before forwarding. With anti-spoofing protection, the gateway router does a source address lookup of the encapsulation packet IP header in the VPN to ensure that legitimate sources are injecting traffic through their designated IP tunnels. This is called the strict mode and is the default behavior of anti-spoofing protection. To pass traffic from nondesignated tunnels, the reverse path forwarding check is enabled in the lose mode. For traffic received from nonexistent sources, the reverse path forwarding check fails for both the strict and loose modes.

Anti-spoofing is supported on VRF routing instances. To enable anti-spoofing for dynamic tunnels, include the `ip-tunnel-rpf-check` statement at the `[edit routing-instances routing-instance-name routing-options forwarding-table]` hierarchy level.

Topology

Figure 76 on page 944 illustrates a sample network topology enabled with anti-spoofing protection. Routers R0, R1 and R2 are each connected to hosts Host0, Host1, and Host2, respectively. Two generic routing encapsulation (GRE) next-hop-based dynamic tunnels, Tunnel 1 and Tunnel 2 – connect Router R0 with Routers R1 and R2, respectively. The VRF routing instance is running between each router and its connected host devices.

Figure 76: Anti-Spoofing Protection for Next-Hop-Based Dynamic Tunnels



Taking as an example, three packets (Packets A, B, and C) are received on Router 0 from Router R2 through the next-hop-based dynamic GRE tunnel (Tunnel 2). The source IP address of these packets are 172.17.0.2 (Packet A), 172.18.0.2 (Packet B), and 172.20.0.2 (Packet C).

The source IP address of Packets A and B belong to Host 2 and Host 1, respectively. Packet C is a nonexistent source tunnel. The designated tunnel in this example is Tunnel 2, and the nondesignated tunnel is Tunnel 1. Therefore, the packets are processed as follows:

- **Packet A**—Because the source is coming from a designated tunnel (Tunnel 2), Packet A passes the reverse path forwarding check and is processed for forwarding through Tunnel 2.

- **Packet B**—Because the source is coming from Tunnel 1, which is a nondesinated tunnel, by default, Packet B fails the reverse path forwarding check in the strict mode. If loose mode is enabled, Packet B is allowed for forwarding.

- **Packet C**—Because the source is a nonexistent tunnel source, Packet C fails the reverse path forwarding check, and the packet is not forwarded.

Configuration

IN THIS SECTION

CLI Quick Configuration

To quickly configure this example, copy the following commands, paste them into a text file, remove any line breaks, change any details necessary to match your network configuration, copy and paste the commands into the CLI at the `[edit]` hierarchy level, and then enter `commit` from configuration mode.

Router R0

```
set interfaces ge-0/0/0 unit 0 family inet address 192.0.2.1/24
set interfaces ge-0/0/1 unit 0 family inet address 198.51.100.1/24
set interfaces ge-0/0/2 vlan-tagging
set interfaces ge-0/0/2 unit 0 vlan-id 1
set interfaces ge-0/0/2 unit 0 family inet address 172.16.0.1/16
set interfaces lo0 unit 0 family inet address 10.1.1.1/32
set routing-options router-id 10.1.1.1
set routing-options autonomous-system 100
set routing-options dynamic-tunnels gre next-hop-based-tunnel
set routing-options dynamic-tunnels T1 source-address 192.0.2.1
set routing-options dynamic-tunnels T1 gre
set routing-options dynamic-tunnels T1 destination-networks 192.0.2.0/24
set routing-options dynamic-tunnels T2 source-address 198.51.100.1
set routing-options dynamic-tunnels T2 gre
set routing-options dynamic-tunnels T2 destination-networks 198.51.100.0/24
set protocols rsvp interface all
set protocols rsvp interface fxp0.0 disable
set protocols bgp group IBGP type internal
set protocols bgp group IBGP local-address 10.1.1.1
set protocols bgp group IBGP family inet-vpn unicast
set protocols bgp group IBGP neighbor 20.1.1.1
set protocols bgp group IBGP neighbor 30.1.1.1
set protocols ospf traffic-engineering
```

```
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set protocols ospf area 0.0.0.0 interface all
set routing-instances VPN1 instance-type vrf
set routing-instances VPN1 interface ge-0/0/2.0
set routing-instances VPN1 route-distinguisher 100:100
set routing-instances VPN1 vrf-target target:100:1
set routing-instances VPN1 vrf-table-label
set routing-instances VPN1 routing-options forwarding-table ip-tunnel-rpf-check mode strict
set routing-instances VPN1 protocols bgp group External type external
set routing-instances VPN1 protocols bgp group External family inet unicast
set routing-instances VPN1 protocols bgp group External peer-as 200
set routing-instances VPN1 protocols bgp group External neighbor 172.16.0.1
```

Router R1

```
set interfaces ge-0/0/0 unit 0 family inet address 192.0.2.2/24
set interfaces ge-0/0/1 vlan-tagging
set interfaces ge-0/0/1 unit 0 vlan-id 2
set interfaces ge-0/0/1 unit 0 family inet address 172.18.0.1/16
set interfaces lo0 unit 0 family inet address 20.1.1.1/32
set routing-options router-id 20.1.1.1
set routing-options autonomous-system 100
set routing-options dynamic-tunnels gre next-hop-based-tunnel
set routing-options dynamic-tunnels T1 source-address 192.0.2.2
set routing-options dynamic-tunnels T1 gre
set routing-options dynamic-tunnels T1 destination-networks 192.0.2.0/24
set protocols rsvp interface all
set protocols rsvp interface fxp0.0 disable
set protocols bgp group IBGP type internal
set protocols bgp group IBGP local-address 20.1.1.1
set protocols bgp group IBGP family inet-vpn unicast
set protocols bgp group IBGP neighbor 30.1.1.1
set protocols bgp group IBGP neighbor 10.1.1.1
set protocols ospf traffic-engineering
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set protocols ospf area 0.0.0.0 interface all
set routing-instances VPN2 instance-type vrf
set routing-instances VPN2 interface ge-0/0/1.0
set routing-instances VPN2 route-distinguisher 100:200
set routing-instances VPN2 vrf-target target:200:1
set routing-instances VPN2 vrf-table-label
```

R2

```
set interfaces ge-0/0/1 unit 0 family inet address 198.51.100.2/24
set interfaces ge-0/0/2 vlan-tagging
set interfaces ge-0/0/2 unit 0 vlan-id 3
set interfaces ge-0/0/2 unit 0 family inet address 172.17.0.1/16
set interfaces lo0 unit 0 family inet address 30.1.1.1/32
set routing-options router-id 30.1.1.1
set routing-options autonomous-system 100
set routing-options dynamic-tunnels gre next-hop-based-tunnel
set routing-options dynamic-tunnels T2 source-address 198.51.100.2
set routing-options dynamic-tunnels T2 gre
set routing-options dynamic-tunnels T2 destination-networks 198.51.100.0/24
set protocols rsvp interface all
set protocols rsvp interface fxp0.0 disable
set protocols bgp group IBGP type internal
set protocols bgp group IBGP local-address 30.1.1.1
set protocols bgp group IBGP family inet-vpn unicast
set protocols bgp group IBGP neighbor 20.1.1.1
set protocols bgp group IBGP neighbor 10.1.1.1
set protocols ospf traffic-engineering
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set protocols ospf area 0.0.0.0 interface all
set routing-instances VPN3 instance-type vrf
set routing-instances VPN3 interface ge-0/0/2.0
set routing-instances VPN3 route-distinguisher 100:300
set routing-instances VPN3 vrf-target target:300:1
set routing-instances VPN3 vrf-table-label
```

Procedure

Step-by-Step Procedure

The following example requires that you navigate various levels in the configuration hierarchy. For information about navigating the CLI, see *Using the CLI Editor in Configuration Mode* in the CLI User Guide.

To configure Router R0:

1. Configure Router R0’s interfaces, including the loopback interface.

```
[edit interfaces]
user@R0# set ge-0/0/0 unit 0 family inet address 192.0.2.1/24
user@R0# set ge-0/0/1 unit 0 family inet address 198.51.100.1/24
user@R0# set ge-0/0/2 vlan-tagging
user@R0# set ge-0/0/2 unit 0 vlan-id 1
user@R0# set ge-0/0/2 unit 0 family inet address 172.16.0.1/16
user@R0# set lo0 unit 0 family inet address 10.1.1.1/32
```

2. Assign the router ID and autonomous system number for Router R0.

```
[edit routing-options]
user@R0# set router-id 10.1.1.1
user@R0# set autonomous-system 100
```

3. Configure IBGP peering between the routers.

```
[edit protocols]
user@R0# set bgp group IBGP type internal
user@R0# set bgp group IBGP local-address 10.1.1.1
user@R0# set bgp group IBGP family inet-vpn unicast
user@R0# set bgp group IBGP neighbor 20.1.1.1
user@R0# set bgp group IBGP neighbor 30.1.1.1
```

4. Configure OSPF on all the interfaces of Router R0, excluding the management interface.

```
[edit protocols]
user@R0# set ospf traffic-engineering
user@R0# set ospf area 0.0.0.0 interface lo0.0 passive
user@R0# set ospf area 0.0.0.0 interface all
```

5. Configure RSVP on all the interfaces of Router R0, excluding the management interface.

```
[edit protocols]
user@R0# set rsvp interface all
user@R0# set rsvp interface fxp0.0 disable
```

6. Enable next-hop-based dynamic GRE tunnel configuration on Router R0.

```
[edit routing-options]
user@R0# set dynamic-tunnels gre next-hop-based-tunnel
```

7. Configure the dynamic GRE tunnel parameters from Router R0 to Router R1.

```
[edit routing-options]
user@R0# set dynamic-tunnels T1 source-address 192.0.2.1
user@R0# set dynamic-tunnels T1 gre
user@R0# set dynamic-tunnels T1 destination-networks 192.0.2.0/24
```

8. Configure the dynamic GRE tunnel parameters from Router R0 to Router R2.

```
[edit routing-options]
user@R0# set dynamic-tunnels T2 source-address 198.51.100.1
user@R0# set dynamic-tunnels T2 gre
user@R0# set dynamic-tunnels T2 destination-networks 198.51.100.0/24
```

9. Configure a virtual routing and forwarding (VRF) routing instance on Router R0, and assign the interface connecting to Host 1 to the VRF instance.

```
[edit routing-instances]
user@R0# set VPN1 instance-type vrf
user@R0# set VPN1 route-distinguisher 100:100
user@R0# set VPN1 vrf-target target:100:1
user@R0# set VPN1 vrf-table-label
user@R0# set VPN1 interface ge-0/0/2.0
```

10. Configure an external BGP session with Host 1 for the VRF routing instance.

```
[edit routing-instances]
user@R0# set VPN1 protocols bgp group External type external
user@R0# set VPN1 protocols bgp group External family inet unicast
user@R0# set VPN1 protocols bgp group External peer-as 200
user@R0# set VPN1 protocols bgp group External neighbor 172.16.0.1
```

11. Configure anti-spoofing protection for the VRF routing instance on Router R0. This enables reverse path forwarding check for the next-hop-based dynamic tunnels, T1 and T2, on Router 0.

    ```
    [edit routing-instances]
    user@R0# set VPN1 routing-options forwarding-table ip-tunnel-rpf-check mode strict
    ```

Results

From configuration mode, confirm your configuration by entering the `show interfaces`, `show routing-options`, `show protocols`, and `show routing-options` commands. If the output does not display the intended configuration, repeat the instructions in this example to correct the configuration.

```
user@R0# show interfaces
ge-0/0/0 {
    unit 0 {
        family inet {
            address 192.0.2.1/24;
        }
    }
}
ge-0/0/1 {
    unit 0 {
        family inet {
            address 198.51.100.1/24;
        }
    }
}
ge-0/0/2 {
    vlan-tagging;
    unit 0 {
        vlan-id 1;
        family inet {
            address 172.16.0.1/16;
        }
    }
}
lo0 {
    unit 0 {
        family inet {
            address 10.1.1.1/32;
        }
```

```
    }
}
```

```
user@R0# show routing-options
router-id 10.1.1.1;
autonomous-system 100;
dynamic-tunnels {
    gre next-hop-based-tunnel;
    T1 {
        source-address 192.0.2.1;
        gre;
        destination-networks {
            192.0.2.0/24;
        }
    }
    T2 {
        source-address 198.51.100.1;
        gre;
        destination-networks {
            198.51.100.0/24;
        }
    }
}
```

```
user@R0# show protocols
rsvp {
    interface all;
    interface fxp0.0 {
        disable;
    }
}
bgp {
    group IBGP {
        type internal;
        local-address 10.1.1.1;
        family inet-vpn {
            unicast;
        }
        neighbor 20.1.1.1;
        neighbor 30.1.1.1;
```

```
    }
}
ospf {
    traffic-engineering;
    area 0.0.0.0 {
        interface lo0.0 {
            passive;
        }
        interface all;
    }
}
```

```
user@R0# show routing-instances
VPN1 {
    instance-type vrf;
    interface ge-0/0/2.0;
    route-distinguisher 100:100;
    vrf-target target:100:1;
    vrf-table-label;
    routing-options {
        forwarding-table {
            ip-tunnel-rpf-check {
                mode strict;
            }
        }
    }
    protocols {
        bgp {
            group External {
                type external;
                family inet {
                    unicast;
                }
                peer-as 200;
                neighbor 172.16.0.1;
            }
        }
    }
}
```

Verification

IN THIS SECTION

Confirm that the configuration is working properly.

Verifying Basic Configuration

Purpose

Verify the OSPF and BGP peering status between the Router R0 and Routers R1 and R2.

Action

From operational mode, run the **show ospf neighbor** and **show bgp summary**commands.

```
user@R0> show ospf neighbor
Address          Interface              State     ID               Pri  Dead
192.0.2.2        ge-0/0/0.0             Full      20.1.1.1         128    32
198.51.100.2     ge-0/0/1.0             Full      30.1.1.1         128    32

user@R0> show bgp summary
Groups: 2 Peers: 3 Down peers: 1
Table          Tot Paths  Act Paths Suppressed    History Damp State    Pending
bgp.l3vpn.0
                       0          0          0          0          0          0
Peer                     AS      InPkt     OutPkt    OutQ   Flaps Last Up/Dwn State|#Active/
Received/Accepted/Damped...
20.1.1.1                100        182        178       0       0     1:20:27 Establ
  bgp.l3vpn.0: 0/0/0/0
30.1.1.1                100        230        225       0       0     1:41:51 Establ
  bgp.l3vpn.0: 0/0/0/0
172.16.0.1              200          0          0       0       0     1:42:08 Establ
```

Meaning

The OSPF and BGP sessions are up and running between the Routers R0, R1, and R2.

Verifying Dynamic Tunnel Configuration

Purpose

Verify the status of the next-hop-based dynamic GRE tunnels between the Router R0 and Routers R1 and R2.

Action

From operational mode, run the **show route table inet.3**, and the **show dynamic-tunnels database terse** commands.

```
user@R0> show route table inet.3

inet.3: 2 destinations, 2 routes (2 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

192.0.2.0/24       *[Tunnel/300] 01:47:57
                      Tunnel
192.0.2.2/24       *[Tunnel/300] 01:47:57
                      Tunnel Composite

198.51.100.0/24    *[Tunnel/300] 01:47:57
                      Tunnel
198.51.100.2/24    *[Tunnel/300] 01:47:57
                      Tunnel Composite
```

```
user@R0> show dynamic-tunnels database terse
Table: inet.3

Destination-network: 192.0.2.0/24
Destination             Source              Next-hop                     Type          Status
192.0.2.2/24      192.0.2.1     0xb395e70 nhid 612         gre       Up

Destination-network: 198.51.100.0/24
Destination             Source                      Next-hop                     Type
```

```
Status
198.51.100.2      198.51.100.1     0xb395e70 nhid 612        gre       Up
```

Meaning

The two next-hop-based dynamic GRE tunnels, Tunnel 1 and Tunnel 2, are up.

Verifying Anti-Spoofing Protection Configuration

Purpose

Verify that the reverse path forwarding check has been enabled on the VRF routing instance on Router R0.

Action

From the operational mode, run the **show krt table VPN1.inet.0 detail**.

```
user@R0> show krt table VPN1.inet.0 detail
KRT tables:
VPN1.inet.0                   : GF: 1 krt-index: 8     ID: 0 kernel-id: 8
        flags: (null)
        tunnel rpf config data : enable, strict, filter [0], 0x2
        tunnel rpf tlv data : enable, strict, filter [0], 0x4
        unicast reverse path: disabled
        fast-reroute-priority: 0
        Permanent NextHops
          Multicast        : 0 Broadcast : 0
          Receive          : 0   Discard   : 0
          Multicast Discard: 0   Reject    : 0
          Local            : 0   Deny      : 0
          Table            : 0
```

Meaning

The configured reverse path forwarding check is enabled on the VRF routing instance in the strict mode.

Change History Table

Feature support is determined by the platform and release you are using. Use Feature Explorer to determine if a feature is supported on your platform.

Release	Description
19.2R1	Starting in Junos Release 19.2R1, on MX Series routers with MPCs and MICs, carrier supporting carrier (CSC) architecture can be deployed with MPLS-over-UDP tunnels carrying MPLS traffic over dynamic IPv4 UDP tunnels that are established between supporting carrier's PE devices.
18.3R1	Starting in Junos OS Release 18.3R1, MPLS-over-UDP tunnels are supported on PTX Series routers and QFX Series switches.
18.2R1	Starting in Junos OS Release 18.2R1, on PTX series routers and QFX10000 with unidirectional MPLS-over-UDP tunnels, you must configure the remote PE device with an input filter for MPLS-over-UDP packets, and an action for decapsulating the IP and UDP headers for forwarding the packets in the reverse tunnel direction.
17.4R1	Starting in Junos OS Release 17.4R1, on MX Series routers, the next-hop-based dynamic GRE tunnels are signaled using BGP encapsulation extended community.
17.4R1	Starting in Junos OS Release 17.4R1, on MX Series routers, the next-hop-based dynamic MPLS-over-UDP tunnels are signaled using BGP encapsulation extended community.
17.1R1	Starting in Junos OS Release 17.1, on MX Series routers with MPCs and MICs, the scaling limit of next-hop-based dynamic GRE tunnels is increased.
17.1R1	Starting in Junos OS Release 17.1, on MX Series routers with MPCs and MICs, the scaling limit of MPLS-over-UDP tunnels is increased.

CHAPTER 7

Protection and Performance Features for Layer 3 VPNs

IN THIS CHAPTER

BGP PIC for Layer 3 VPNs

IN THIS SECTION

Configuring BGP PIC Edge for MPLS Layer 3 VPNs

In an MPLS VPN Layer 3 environment, it is common for customers to multihome their networks to provide link redundancy. Although the interior gateway protocol (IGP) can provide fast convergence, in certain instances, the time to resolve a link failure and provide an alternate route can be time consuming. For example, a provider edge (PE) router might be configured with 200,000 or more IP prefixes, and a PE router failure could affect many of those prefixes.

BGP Prefix-Independent Convergence (PIC) Edge allows you to install a Layer 3 VPN route in the forwarding table as an alternate path, enabling fast failover when a PE router fails or you lose connectivity to a PE router. This already installed path is used until global convergence through the IGP is resolved. Using the alternative VPN route for forwarding until global convergence is complete reduces traffic loss.

BGP PIC Edge supports multiprotocol BGP IPv4 or IPv6 VPN network layer reachability information (NLRI) resolved using any of these IGP protocols:

- OSPF
- IS-IS
- LDP
- RSVP

BGP PIC Edge does not support multicast traffic.

Before you begin:

1. Configure LDP or RSVP.
2. Configure an IGP: either OSPF or IS-IS.
3. Configure a Layer 3 VPN.
4. Configure multiprotocol BGP for either an IPv4 VPN or an IPv6 VPN.

To configure BGP PIC Edge in an MPLS Layer 3 VPN:

1. Enable BGP PIC Edge:

```
[edit routing-instances routing-instance-name routing-options]
user@host# set protect core
```

NOTE: The BGP PIC edge feature is supported on ACX Universal Metro routers and on MX Series 5G Universal Routing Platforms with MPC interfaces.

2. Configure per-packet load balancing:

```
[edit policy-options]
user@host# set policy-statement policy-name then load-balance per-packet
```

3. Apply the per-packet load balancing policy to routes exported from the routing table to the forwarding table:

```
[edit routing-options forwarding-table]
user@host# set export policy-statement-name
```

4. Verify that BGP PIC Edge is working.

   From operational mode, enter the `show route extensive` command:

```
user@host> show route 192.0.2.6 extensive
ed.inet.0: 6 destinations, 9 routes (6 active, 0 holddown, 0 hidden)
  192.0.2.6/24 (3 entries, 2 announced)
          State: <CalcForwarding>
  TSI:
  KRT in-kernel 192.0.2.6/24 -> {indirect(1048574), indirect(1048577)}
  Page 0 idx 0 Type 1 val 9219e30
      Nexthop: Self
      AS path: [2] 3 I
      Communities: target:2:1
  Path 192.0.2.6 from 192.0.2.4 Vector len 4.  Val: 0
..
          #Multipath Preference: 255
                  Next hop type: Indirect
                  Address: 0x93f4010
                  Next-hop reference count: 2
..
                  Protocol next hop: 192.0.2.4
                  Push 299824
                  Indirect next hop: 944c000 1048574 INH Session ID: 0x3
                  Indirect next hop: weight 0x1
                  Protocol next hop: 192.0.2.5
                  Push 299824
                  Indirect next hop: 944c1d8 1048577 INH Session ID: 0x4
                  Indirect next hop: weight 0x4000
                  State: <ForwardingOnly Int Ext)>
                  Inactive reason: Forwarding use only
                  Age: 25         Metric2: 15
                  Validation State: unverified
                  Task: RT
                  Announcement bits (1): 0-KRT
```

```
                    AS path: 3 I
                    Communities: target:2:1
```

The output lines that contain `Indirect next hop: weight` follow next hops that the software can use to repair paths where a link failure occurs. The next-hop weight has one of the following values:

- 0x1 indicates active next hops.
- 0x4000 indicates passive next hops.

> **BEST PRACTICE**: On MX Series 5G Universal Routing Platforms with Modular Port Concentrators (MPCs), we strongly recommend that you enable enhanced IP network services.
>
> To enable enhanced IP network services:
>
> ```
> [edit chassis]
> user@host# set network-services enhanced-ip
> ```

Example: Configuring BGP PIC Edge for MPLS Layer 3 VPNs

IN THIS SECTION

This example shows how to configure BGP prefix-independent convergence (PIC) edge, which allows you to install a Layer 3 VPN route in the forwarding table as an alternate path. This enables fast failover when a provider edge (PE) router fails or you lose connectivity to a PE router. This already installed path is used until global convergence through the interior gateway protocol (IGP) is resolved. Using the alternative VPN route for forwarding until global convergence is complete reduces traffic loss.

Requirements

No special configuration beyond device initialization is required before configuring this example.

This example uses the following hardware and software components:

- One MX Series 5G Universal Routing Platforms with MPC interfaces to configure the BGP PIC edge feature.
- Five routers that can be a combination of M Series Multiservice Edge Routers, MX Series 5G Universal Routing Platforms, or T Series Core Routers.
- Junos OS Release 13.2 or later on the device with BGP PIC edge configured.

Overview

IN THIS SECTION

In an MPLS VPN Layer 3 environment, it is common for customers to multihome their networks to provide link redundancy. Although the interior gateway protocol (IGP) can provide fast convergence, in certain instances, the time to resolve a link failure and provide an alternate route can be time consuming. For example, a provider edge (PE) router might be configured with 200,000 or more IP prefixes, and a PE router failure could affect many of those prefixes.

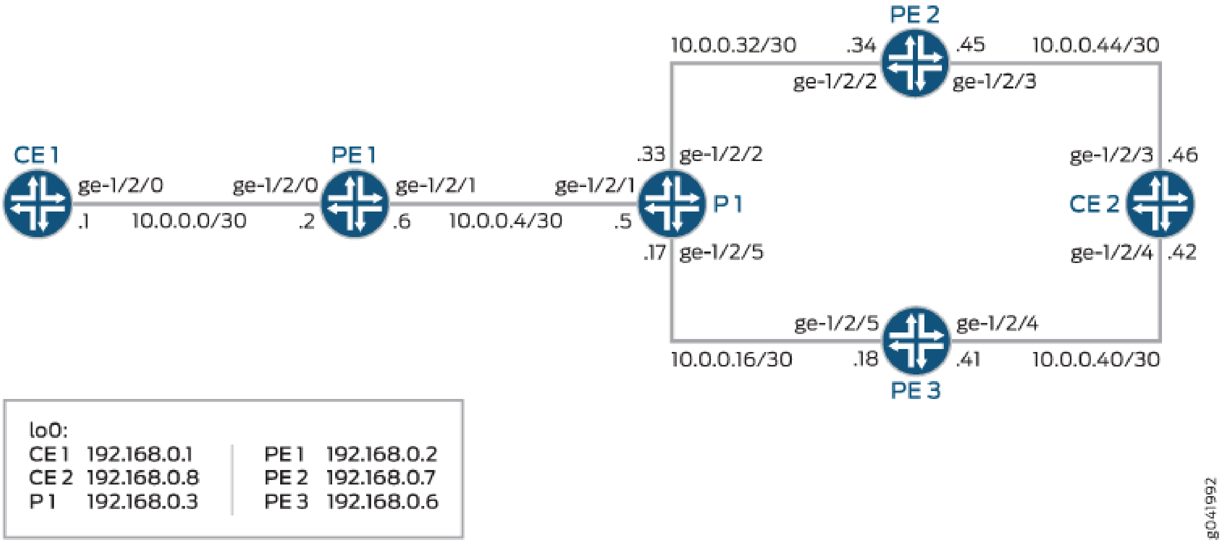
This example shows two customer edge (CE) routers, Device CE1 and Device CE2. Devices PE1, PE2, and PE3 are PE routers. Device P1 is a provider core router. Only Device PE1 has BGP PIC edge configured. The example uses the P1-PE2 link (P-PE) link to simulate the loss of a section of the network.

For testing, the address 172.16.1.5/24 is added as a loopback interface address on Device CE2. The address is announced to Device PE2 and Device PE3 and is relayed by way of internal BGP (IBGP) IBGP to Device PE1. On Device PE1, there are two paths to the 172.16.1.5/24 network. These are the primary and a backup path.

Topology

Figure 77 on page 962 shows the sample network.

Figure 77: BGP PIC Edge Scenario



"CLI Quick Configuration" on page 962 shows the configuration for all of the devices in Figure 77 on page 962.

The section "Step-by-Step Procedure" on page 966 describes the steps on Device PE1.

Configuration

IN THIS SECTION

CLI Quick Configuration

To quickly configure this example, copy the following commands, paste them into a text file, remove any line breaks, change any details necessary to match your network configuration, and then copy and paste the commands into the CLI at the `[edit]` hierarchy level.

Device CE1

```
set interfaces ge-1/2/0 unit 0 family inet address 10.0.0.1/30
set interfaces lo0 unit 0 family inet address 192.168.0.1/32
set protocols bgp group ebgp type external
```

```
set protocols bgp group ebgp export send-direct
set protocols bgp group ebgp neighbor 10.0.0.2
set policy-options policy-statement send-direct from protocol direct
set policy-options policy-statement send-direct then accept
set routing-options autonomous-system 101
```

Device CE2

```
set interfaces ge-1/2/4 unit 0 family inet address 10.0.0.42/30
set interfaces ge-1/2/3 unit 0 family inet address 10.0.0.46/30
set interfaces lo0 unit 0 family inet address 192.168.0.8/32
set interfaces lo0 unit 0 family inet address 172.16.1.5/24
set protocols bgp group ebgp type external
set protocols bgp group ebgp export send-direct
set protocols bgp group ebgp neighbor 10.0.0.45
set protocols bgp group ebgp neighbor 10.0.0.41
set policy-options policy-statement send-direct from protocol direct
set policy-options policy-statement send-direct then accept
set routing-options autonomous-system 102
```

Device P1

```
set interfaces ge-1/2/1 unit 0 family inet address 10.0.0.5/30
set interfaces ge-1/2/1 unit 0 family mpls
set interfaces ge-1/2/5 unit 0 family inet address 10.0.0.17/30
set interfaces ge-1/2/5 unit 0 family mpls
set interfaces ge-1/2/2 unit 0 family inet address 10.0.0.33/30
set interfaces ge-1/2/2 unit 0 family mpls
set interfaces lo0 unit 0 family inet address 192.168.0.3/32
set protocols mpls interface ge-1/2/1.0
set protocols mpls interface ge-1/2/5.0
set protocols mpls interface ge-1/2/2.0
set protocols ospf area 0.0.0.0 interface ge-1/2/1.0
set protocols ospf area 0.0.0.0 interface ge-1/2/5.0
set protocols ospf area 0.0.0.0 interface ge-1/2/2.0
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set protocols ldp interface ge-1/2/1.0
set protocols ldp interface ge-1/2/5.0
set protocols ldp interface ge-1/2/2.0
set protocols ldp interface lo0.0
set routing-options autonomous-system 100
```

Device PE1

```
set interfaces ge-1/2/0 unit 0 family inet address 10.0.0.2/30
set interfaces ge-1/2/1 unit 0 family inet address 10.0.0.6/30
set interfaces ge-1/2/1 unit 0 family mpls
set interfaces lo0 unit 0 family inet address 192.168.0.2/32
set protocols mpls interface ge-1/2/1.0
set protocols bgp group ibgp type internal
set protocols bgp group ibgp local-address 192.168.0.2
set protocols bgp group ibgp family inet unicast
set protocols bgp group ibgp family inet-vpn unicast
set protocols bgp group ibgp export nhs
set protocols bgp group ibgp neighbor 192.168.0.7
set protocols bgp group ibgp neighbor 192.168.0.6
set protocols ospf area 0.0.0.0 interface ge-1/2/1.0
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set protocols ldp interface ge-1/2/1.0
set protocols ldp interface lo0.0
set policy-options policy-statement lb then load-balance per-packet
set policy-options policy-statement nhs then next-hop self
set routing-instances customer1 instance-type vrf
set routing-instances customer1 interface ge-1/2/0.0
set routing-instances customer1 route-distinguisher 100:1
set routing-instances customer1 vrf-target target:100:1
set routing-instances customer1 routing-options protect core
set routing-instances customer1 protocols bgp group ebgp type external
set routing-instances customer1 protocols bgp group ebgp neighbor 10.0.0.1
set routing-options router-id 192.168.0.2
set routing-options autonomous-system 100
set routing-options forwarding-table export lb
```

Device PE2

```
set interfaces ge-1/2/2 unit 0 family inet address 10.0.0.34/30
set interfaces ge-1/2/2 unit 0 family mpls
set interfaces ge-1/2/3 unit 0 family inet address 10.0.0.45/30
set interfaces lo0 unit 0 family inet address 192.168.0.7/32
set protocols mpls interface ge-1/2/2.0
set protocols bgp group ibgp type internal
set protocols bgp group ibgp local-address 192.168.0.7
set protocols bgp group ibgp family inet unicast
```

```
set protocols bgp group ibgp family inet-vpn unicast
set protocols bgp group ibgp export nhs
set protocols bgp group ibgp neighbor 192.168.0.2
set protocols bgp group ibgp neighbor 192.168.0.6
set protocols ospf area 0.0.0.0 interface ge-1/2/2.0
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set protocols ldp interface ge-1/2/2.0
set protocols ldp interface lo0.0
set routing-instances customer1 instance-type vrf
set routing-instances customer1 interface ge-1/2/3.0
set routing-instances customer1 route-distinguisher 100:1
set routing-instances customer1 vrf-target target:100:1
set routing-instances customer1 protocols bgp group ebgp type external
set routing-instances customer1 protocols bgp group ebgp neighbor 10.0.0.46
set routing-options autonomous-system 100
```

Device PE3

```
set interfaces ge-1/2/5 unit 0 family inet address 10.0.0.18/30
set interfaces ge-1/2/5 unit 0 family mpls
set interfaces ge-1/2/4 unit 0 family inet address 10.0.0.41/30
set interfaces ge-1/2/4 unit 0 family mpls
set interfaces lo0 unit 0 family inet address 192.168.0.6/32
set protocols mpls interface ge-1/2/5.0
set protocols mpls interface ge-1/2/4.0
set protocols bgp group ibgp type internal
set protocols bgp group ibgp local-address 192.168.0.6
set protocols bgp group ibgp family inet unicast
set protocols bgp group ibgp family inet-vpn unicast
set protocols bgp group ibgp export nhs
set protocols bgp group ibgp neighbor 192.168.0.7
set protocols bgp group ibgp neighbor 192.168.0.2
set protocols ospf area 0.0.0.0 interface ge-1/2/5.0
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set protocols ldp interface ge-1/2/5.0
set protocols ldp interface lo0.0
set routing-instances customer1 instance-type vrf
set routing-instances customer1 interface ge-1/2/4.0
set routing-instances customer1 route-distinguisher 100:1
set routing-instances customer1 vrf-target target:100:1
set routing-instances customer1 protocols bgp group ebgp type external
```

```
set routing-instances customer1 protocols bgp group ebgp neighbor 10.0.0.42
set routing-options autonomous-system 100
```

Procedure

Step-by-Step Procedure

The following example requires that you navigate various levels in the configuration hierarchy. For information about navigating the CLI, see *Using the CLI Editor in Configuration Mode* in the CLI User Guide.

To configure Device R1:

1. Configure the device interfaces.

   ```
   [edit interfaces]
   user@PE1# set ge-1/2/0 unit 0 family inet address 10.0.0.2/30
   user@PE1# set ge-1/2/1 unit 0 family inet address 10.0.0.6/30
   user@PE1# set ge-1/2/1 unit 0 family mpls
   user@PE1# set lo0 unit 0 family inet address 192.168.0.2/32
   ```

2. Configure MPLS and LDP on the core-facing interfaces.

   ```
   [edit protocols]
   user@PE1# set mpls interface ge-1/2/1.0
   user@PE1# set ldp interface ge-1/2/1.0
   user@PE1# set ldp interface lo0.0
   ```

3. Configure an IGP on the core-facing interfaces.

   ```
   [edit protocols ospf area 0.0.0.0]
   user@PE1# set interface ge-1/2/1.0
   user@PE1# set interface lo0.0 passive
   ```

4. Configure IBGP connections with the other PE devices.

   ```
   [edit protocols bgp group ibgp]
   user@PE1# set type internal
   user@PE1# set local-address 192.168.0.2
   ```

```
user@PE1# set family inet unicast
user@PE1# set family inet-vpn unicast
user@PE1# set export nhs
user@PE1# set neighbor 192.168.0.7
user@PE1# set neighbor 192.168.0.6
```

5. Configure the load-balancing policy.

```
[edit policy-options policy-statement lb]
user@PE1# set then load-balance per-packet
```

6. (Optional) Configure a next-hop self policy.

```
[edit policy-options policy-statement nhs]
user@PE1# set then next-hop self
```

7. Configure the routing-instance to create the CE-PE EBGP connection.

```
[edit routing-instances customer1]
user@PE1# set instance-type vrf
user@PE1# set interface ge-1/2/0.0
user@PE1# set route-distinguisher 100:1
user@PE1# set vrf-target target:100:1
user@PE1# set protocols bgp group ebgp type external
user@PE1# set protocols bgp group ebgp neighbor 10.0.0.1
```

8. Enable the BGP PIC edge feature.

```
[edit routing-instances customer1]
user@PE1# set  routing-options protect core
```

9. Apply the load-balancing policy.

```
[edit routing-options forwarding-table]
user@PE1# set export lb
```

10. Assign the router ID and autonomous system (AS) number.

```
[edit routing-options]
user@PE1# set router-id 192.168.0.2
user@PE1# set autonomous-system 100
```

Results

From configuration mode, confirm your configuration by entering the `show interfaces`, `show protocols`, `show policy-options`, `show routing-instances`, and `show routing-options` commands. If the output does not display the intended configuration, repeat the instructions in this example to correct the configuration.

```
user@PE1# show interfaces
ge-1/2/0 {
    unit 0 {
        family inet {
            address 10.0.0.2/30;
        }
    }
}
ge-1/2/1 {
    unit 0 {
        family inet {
            address 10.0.0.6/30;
        }
        family mpls;
    }
}
lo0 {
    unit 0 {
        family inet {
            address 192.168.0.2/32;
        }
    }
}
```

```
user@PE1# show protocols
mpls {
    interface ge-1/2/1.0;
```

```
}
bgp {
    group ibgp {
        type internal;
        local-address 192.168.0.2;
        family inet {
            unicast;
        }
        family inet-vpn {
            unicast;
        }
        export nhs;
        neighbor 192.168.0.7;
        neighbor 192.168.0.6;
    }
}
ospf {
    area 0.0.0.0 {
        interface ge-1/2/1.0;
        interface lo0.0 {
            passive;
        }
    }
}
ldp {
    interface ge-1/2/1.0;
    interface lo0.0;
}
```

```
user@PE1# show policy-options
policy-statement lb {
    then {
        load-balance per-packet;
    }
}
policy-statement nhs {
    then {
        next-hop self;
```

```
    }
}
```

```
user@PE1# show routing-instances
customer1 {
    instance-type vrf;
    interface ge-1/2/0.0;
    route-distinguisher 100:1;
    vrf-target target:100:1;
    routing-options {
        protect core;
    }
    protocols {
        bgp {
            group ebgp {
                type external;
                peer-as 101;
                neighbor 10.0.0.1;
            }
        }
    }
}
```

```
user@PE1# show routing-options
router-id 192.168.0.2;
autonomous-system 100;
forwarding-table {
    export lb;
}
```

If you are done configuring the device, enter `commit` from configuration mode.

Verification

IN THIS SECTION

Confirm that the configuration is working properly.

Displaying Extensive Route Information

Purpose

Confirm that BGP PIC Edge is working.

Action

From Device PE1, run the `show route extensive table customer1.inet.0 172.16.1/24` command.

```
user@PE1> show route extensive table customer1.inet.0 172.16.1/24

customer1.inet.0: 7 destinations, 12 routes (7 active, 0 holddown, 0 hidden)
172.16.1.0/24 (3 entries, 2 announced)
        State: <CalcForwarding>
TSI:
KRT in-kernel 172.16.1.0/24 -> {indirect(262146), indirect(262142)}
Page 0 idx 0, (group ebgp type External) Type 1 val 0x950a62c (adv_entry)
   Advertised metrics:
     Nexthop: Self
     AS path: [100] 102 I
     Communities: target:100:1
Path 172.16.1.0 from 192.168.0.6 Vector len 4.  Val: 0
        @BGP    Preference: 170/-101
                Route Distinguisher: 100:1
                Next hop type: Indirect
                Address: 0x9514a74
                Next-hop reference count: 7
                Source: 192.168.0.6
                Next hop type: Router, Next hop index: 990
                Next hop: 10.0.0.5 via ge-1/2/1.0, selected
                Label operation: Push 299824, Push 299856(top)
                Label TTL action: prop-ttl, prop-ttl(top)
```

```
               Load balance label: Label 299824: None; Label 299856: None;
               Session Id: 0x280002
               Protocol next hop: 192.168.0.6
               Label operation: Push 299824
               Label TTL action: prop-ttl
               Load balance label: Label 299824: None;
               Indirect next hop: 0x96bc104 262146 INH Session ID: 0x280006
               State: <Secondary Active Int Ext ProtectionPath ProtectionCand>
               Local AS:   100 Peer AS:   100
               Age: 1:38:13    Metric2: 1
               Validation State: unverified
               Task: BGP_100.192.168.0.6+45824
               Announcement bits (1): 1-BGP_RT_Background
               AS path: 102 I
               Communities: target:100:1
               Import Accepted
               VPN Label: 299824
               Localpref: 100
               Router ID: 192.168.0.6
               Primary Routing Table bgp.l3vpn.0
               Indirect next hops: 1
                       Protocol next hop: 192.168.0.6 Metric: 1
                       Label operation: Push 299824
                       Label TTL action: prop-ttl
                       Load balance label: Label 299824: None;
                       Indirect next hop: 0x96bc104 262146 INH Session ID: 0x280006
                       Indirect path forwarding next hops: 1
                               Next hop type: Router
                               Next hop: 10.0.0.5 via ge-1/2/1.0
                               Session Id: 0x280002
                       192.168.0.6/32 Originating RIB: inet.3
                         Metric: 1                       Node path count: 1
                         Forwarding nexthops: 1
                               Nexthop: 10.0.0.5 via ge-1/2/1.0
        BGP    Preference: 170/-101
               Route Distinguisher: 100:1
               Next hop type: Indirect
               Address: 0x9515570
               Next-hop reference count: 7
               Source: 192.168.0.7
               Next hop type: Router, Next hop index: 933
               Next hop: 10.0.0.5 via ge-1/2/1.0, selected
               Label operation: Push 299856, Push 299872(top)
```

```
                Label TTL action: prop-ttl, prop-ttl(top)
                Load balance label: Label 299856: None; Label 299872: None;
                Session Id: 0x280002
                Protocol next hop: 192.168.0.7
                Label operation: Push 299856
                Label TTL action: prop-ttl
                Load balance label: Label 299856: None;
                Indirect next hop: 0x96bc000 262142 INH Session ID: 0x280005
                State: <Secondary NotBest Int Ext ProtectionPath ProtectionCand>
                Inactive reason: Not Best in its group - Router ID
                Local AS:   100 Peer AS:   100
                Age: 1:38:13    Metric2: 1
                Validation State: unverified
                Task: BGP_100.192.168.0.7+10985
                AS path: 102 I
                Communities: target:100:1
                Import Accepted
                VPN Label: 299856
                Localpref: 100
                Router ID: 192.168.0.7
                Primary Routing Table bgp.l3vpn.0
                Indirect next hops: 1
                        Protocol next hop: 192.168.0.7 Metric: 1
                        Label operation: Push 299856
                        Label TTL action: prop-ttl
                        Load balance label: Label 299856: None;
                        Indirect next hop: 0x96bc000 262142 INH Session ID: 0x280005
                        Indirect path forwarding next hops: 1
                                Next hop type: Router
                                Next hop: 10.0.0.5 via ge-1/2/1.0
                                Session Id: 0x280002
                        192.168.0.7/32 Originating RIB: inet.3
                          Metric: 1                       Node path count: 1
                          Forwarding nexthops: 1
                                Nexthop: 10.0.0.5 via ge-1/2/1.0
        #Multipath Preference: 255
                Next hop type: Indirect
                Address: 0x9578010
                Next-hop reference count: 4
                Next hop type: Router, Next hop index: 990
                Next hop: 10.0.0.5 via ge-1/2/1.0, selected
                Label operation: Push 299824, Push 299856(top)
                Label TTL action: prop-ttl, prop-ttl(top)
```

```
                Load balance label: Label 299824: None; Label 299856: None;
                Session Id: 0x280002
                Next hop type: Router, Next hop index: 933
                Next hop: 10.0.0.5 via ge-1/2/1.0
                Label operation: Push 299856, Push 299872(top)
                Label TTL action: prop-ttl, prop-ttl(top)
                Load balance label: Label 299856: None; Label 299872: None;
                Session Id: 0x280002
                Protocol next hop: 192.168.0.6
                Label operation: Push 299824
                Label TTL action: prop-ttl
                Load balance label: Label 299824: None;
                Indirect next hop: 0x96bc104 262146 INH Session ID: 0x280006 Weight 0x1
                Protocol next hop: 192.168.0.7
                Label operation: Push 299856
                Label TTL action: prop-ttl
                Load balance label: Label 299856: None;
                Indirect next hop: 0x96bc000 262142 INH Session ID: 0x280005 Weight 0x4000
                State: <ForwardingOnly Int Ext>
                Inactive reason: Forwarding use only
                Age: 1:38:13    Metric2: 1
                Validation State: unverified
                Task: RT
                Announcement bits (1): 0-KRT
                AS path: 102 I
                Communities: target:100:1
```

Meaning

The Indirect next hop output lines that contain weight follow next hops that the software can use to repair paths where a link failure occurs.

The next-hop weight has one of the following values:

- 0x1 indicates active next hops.
- 0x4000 indicates passive next hops.

Displaying the Forwarding Table

Purpose

Check the forwarding and kernel routing-table state by using `show route forwarding-table`.

Action

From Device PE1, run the `show route forwarding-table table customer1 destination 172.16.1.0/24` command.

```
user@PE1> show route forwarding-table table customer1 destination 172.16.1.0/24

Routing table: customer1.inet
Internet:
Destination        Type RtRef Next hop           Type Index    NhRef Netif
172.16.1.0/24      user     0                    ulst   262147     2
                                                 indr   262146     3
                              10.0.0.5          Push 299824, Push 299856(top)      990     2
ge-1/2/1.0
                                                 indr   262144     3
                              10.0.0.5          Push 300080, Push 299920(top)     1000     2
ge-1/2/1.0
```

Meaning

in addition to the forwarding and kernel routing-table state, this command shows the unilist index (262147) used by the Packet Forwarding Engine.

Displaying the OSPF Routes

Purpose

Show the OSPF route state.

Action

From Device PE1, run the `show (ospf | ospf3) route detail` command.

```
user@PE1> show ospf route detail
```

```
betsy@tp0:PE1> show ospf route detail
Topology default Route Table:

Prefix             Path  Route      NH       Metric NextHop       Nexthop
                   Type  Type       Type            Interface     Address/LSP
192.168.0.3        Intra Router     IP            1 ge-1/2/1.0    10.0.0.5
  area 0.0.0.0, origin 192.168.0.3, optional-capability 0x0
192.168.0.6        Intra Router     IP            2 ge-1/2/1.0    10.0.0.5
  area 0.0.0.0, origin 192.168.0.6, optional-capability 0x0
192.168.0.7        Intra Router     IP            2 ge-1/2/1.0    10.0.0.5
  area 0.0.0.0, origin 192.168.0.7, optional-capability 0x0
10.0.0.4/30        Intra Network    IP            1 ge-1/2/1.0
  area 0.0.0.0, origin 192.168.0.3, priority low
10.0.0.16/30       Intra Network    IP            2 ge-1/2/1.0    10.0.0.5
  area 0.0.0.0, origin 192.168.0.6, priority medium
10.0.0.32/30       Intra Network    IP            2 ge-1/2/1.0    10.0.0.5
  area 0.0.0.0, origin 192.168.0.7, priority medium
192.168.0.2/32     Intra Network    IP            0 lo0.0
  area 0.0.0.0, origin 192.168.0.2, priority low
192.168.0.3/32     Intra Network    IP            1 ge-1/2/1.0    10.0.0.5
  area 0.0.0.0, origin 192.168.0.3, priority medium
192.168.0.6/32     Intra Network    IP            2 ge-1/2/1.0    10.0.0.5
  area 0.0.0.0, origin 192.168.0.6, priority medium
  session-id: 2621446, version: 1
192.168.0.7/32     Intra Network    IP            2 ge-1/2/1.0    10.0.0.5
  area 0.0.0.0, origin 192.168.0.7, priority medium
  session-id: 2621450, version: 1
```

Meaning

The output shows the tracked session IDs for the loopback interface addresses on Devices PE2 and PE3.

RELATED DOCUMENTATION

Example: Configuring Provider Edge Link Protection in Layer 3 VPNs | **1068**

Example: Load Balancing BGP Traffic

Network Services Mode Overview

Firewall Filters and Enhanced Network Services Mode Overview

Configuring Junos OS to Run a Specific Network Services Mode in MX Series Routers

Configuring Enhanced IP Network Services for a Virtual Chassis

Egress Protection in Layer 3 VPNs

IN THIS SECTION

This topic introduces the concept and components in egress protection in layer 3 VPN. It describes and provides examples on how to configure the protected, protector, and point of local repair (PLR) routers.

Egress Protection for BGP Labeled Unicast

When network node or link failures occur, it takes some time to restore service using traditional routing table convergence. Local repair procedures can provide much faster restoration by establishing local protection as close to a failure as possible. Fast protection for egress nodes is available to services in which BGP labeled unicast interconnects IGP areas, levels, or autonomous systems (ASs). If a provider router detects that an egress router (AS or area border router) is down, it immediately forwards the traffic destined to that router to a protector router that forwards the traffic downstream to the destination.

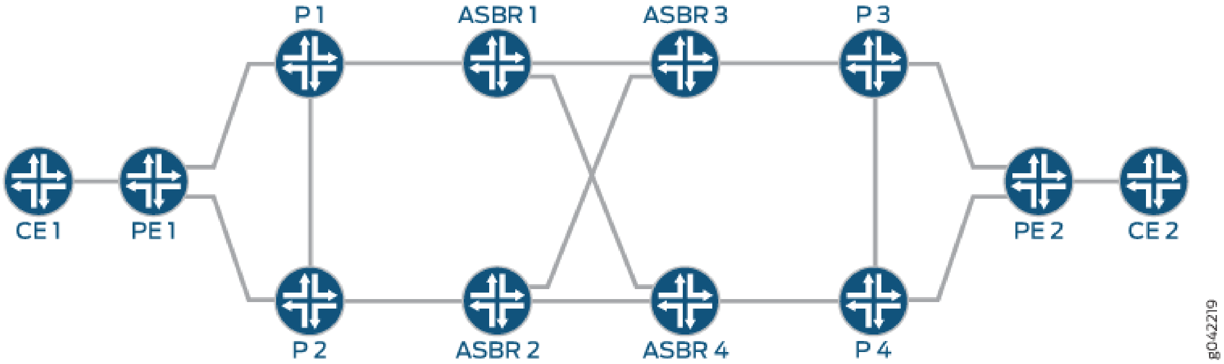
To provide egress protection for BGP labeled unicast, the protector node must create a backup state for downstream destinations before the failure happens. The basic idea of the solution is that the protector node constructs a forwarding state associated with the protected node and relays the MPLS labels assigned by the protected node further downstream to the final destination.

This feature supports the applications Inter-AS Option C and Seamless MPLS.

Inter-AS Option C—BGP labeled unicast provides end-to-end transport label-switched paths (LSPs) by stitching the intra-AS LSPs together. AS boundary routers run EBGP to other AS boundary routers to exchange labels for /32 PE loopback routes. IBGP runs between the provider edge router and AS boundary routers within each AS. In Figure 78 on page 978, the traffic goes from CE1 to CE2. ASBR1 is the protected AS boundary router, ASBR2 is the protector, and Device P1 is the point of local repair (PLR). The primary path is chosen from PE1 to PE2 over ASBR1 and ASBR3. When ASBR1 fails, Router

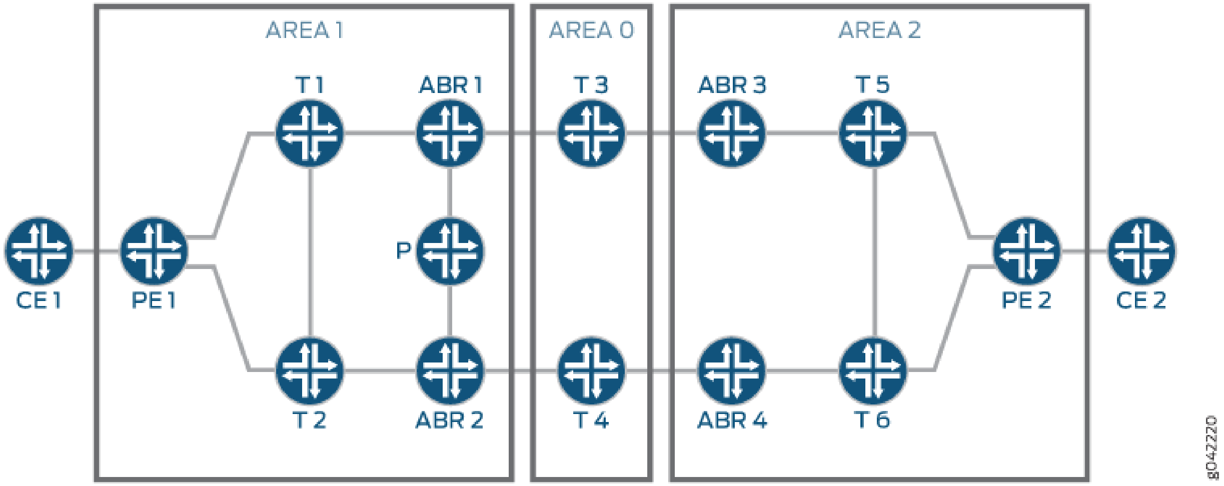
P1 detects the ASBR1 failure and forwards the traffic to ASBR2, which provides backup service and forwards the traffic downstream.

Figure 78: Inter-AS Option C



Seamless MPLS—BGP labeled unicast provides end-to-end transport LSPs by stitching the intra-area/level LSPs. Area border routers (ABRs) run BGP labeled unicast to other ABRs to exchange labels for /32 PE loopback routes. In Figure 79 on page 978, the traffic goes from Device CE1 to Device CE2. ABR1 is the protected ABR, ABR2 is the protector, and T1 is the PLR. The primary path is chosen from PE1 to PE2 over ABR1 and ABR3. When ABR1 fails, Router T1 detects the ABR1 failure and forwards the traffic to ABR2, which provides backup service and forwards the traffic downstream.

Figure 79: Seamless MPLS



In each of these applications, the protected node advertises a primary BGP labeled unicast route that needs protection. When fast protection is enabled, BGP advertises the label routes with a special address as the next hop. This special address is a context identifier that is configured through the CLI.

The protected node also advertises the context identifier in IGP and a NULL label in LDP for the context identifier.

The backup node advertises backup BGP labeled unicast routes for the protected routes. The protector node forwards traffic to the backup node using the labels advertised by the backup node.

The protector node provides the backup service by cross-connecting the labels originated by the protected node and the labels originated by the backup node. The protector node forwards the traffic to the backup node in case of failure of the protected node. The protector node advertises the same context-identifier into IGP with high metric. Also, it advertises a real label in LDP for the context identifier. The protector node listens for the BGP labeled unicast routes advertised by both the protected node and backup node and populates the context label table and backup FIB. When traffic with the real context LDP label arrives, the lookup is done in the context of a protected node. The protector node often acts as the backup node.

The PLR detects the protected node failure and forwards the MPLS traffic to the protector node. The high IGP metric along with the LDP label advertised by the protector node ensure that the PLR uses the protector node as an LDP backup LSP.

There are two supported protection types: collocated protector and centralized protector. In the collocated type, the protector node is also the backup node. In the centralized type, the backup node is different from the protector node.

Configuring Egress Protection for BGP Labeled Unicast

Fast protection for egress nodes is available to services in which BGP labeled unicast interconnects IGP areas, levels, or ASs. If a provider router detects that an egress router (AS or area border router) is down, it immediately forwards the traffic destined to that router to a protector router that forwards the traffic downstream to the destination.

Before configuring egress protection for BGP labeled unicast, ensure that all routers in the AS or area are running Junos OS 14.1 or a later release.

To configure egress protection for BGP labeled unicast:

1. Add the following configuration to the *protected* router:

```
[edit protocols]
       mpls {
           egress-protection {
               context-identifier context-id {
                    primary;
               }
           }
       }
```

```
        bgp {
            group group-name {
                type internal;
                family inet {
                    labeled-unicast {
                        egress-protection {
                            context-identifier context-id;
                        }
                    }
                }
            }
         }
```

2. Add the following configuration to the *protector* router:

```
[edit protocols]
       mpls {
           egress-protection {
               context-identifier context-id {
                    protector;
               }
           }
       }
       bgp {
           group group-name {
               type internal;
               family inet {
                   labeled-unicast {
                       egress-protection;
                   }
               }
           }
        }
```

3. Add the following configuration to the *PLR* (point of local repair) router:

```
[edit protocols]
mpls {
    interface all;
    interface fxp0.0 {
        disable;
    }
```

```
}
isis {
    backup-spf-options per-prefix-calculation;
    level 1 disable;
    interface all {
        node-link-protection;
    }
}
ldp {
    track-igp-metric;
    interface all;
    interface fxp0.0 {
        disable;
    }
}
```

4. Run `show bgp neighbor` on the protected router to verify that egress protection is enabled, for example:

```
user@host# run show bgp neighbor
Peer: 192.0.2.2+179 AS 65536 Local: 192.0.2.1+59264 AS 65536
Type: Internal    State: Established    Flags: <ImportEval Sync>
Last State: OpenConfirm   Last Event: RecvKeepAlive
Last Error: None
Options: <Preference LocalAddress KeepAll AddressFamily Rib-group Refresh>
Address families configured: inet-label-unicast
Local Address: 192.0.2.1 Holdtime: 90 Preference: 170
NLRI configured with egress-protection: inet-label-unicast
Egress-protection NLRI inet-label-unicast
Number of flaps: 0
```

SEE ALSO

egress-protection (MPLS)

Example: Configuring Egress Protection for BGP Labeled Unicast

IN THIS SECTION

This example shows how to configure BGP labeled unicast protection that can be used in case of a PE failure in an Inter-AS Option C topology.

Requirements

This example uses the following hardware and software components:

- M Series Multiservice Edge Routers, MX Series 5G Universal Routing Platforms, or T Series Core Routers
- Junos OS Release 14.1 or later

Overview

IN THIS SECTION

When network node or link failures occur, it takes some time to restore service using traditional routing table convergence. Local repair procedures can provide much faster restoration by establishing local protection as close to a failure as possible. Fast protection for egress nodes is available to services in which BGP labeled unicast interconnects IGP areas, levels, or autonomous systems (ASs). If a provider router detects that an egress router (AS or area border router) is down, it immediately forwards the traffic destined to that router to a protector router that forwards the traffic downstream to the destination.

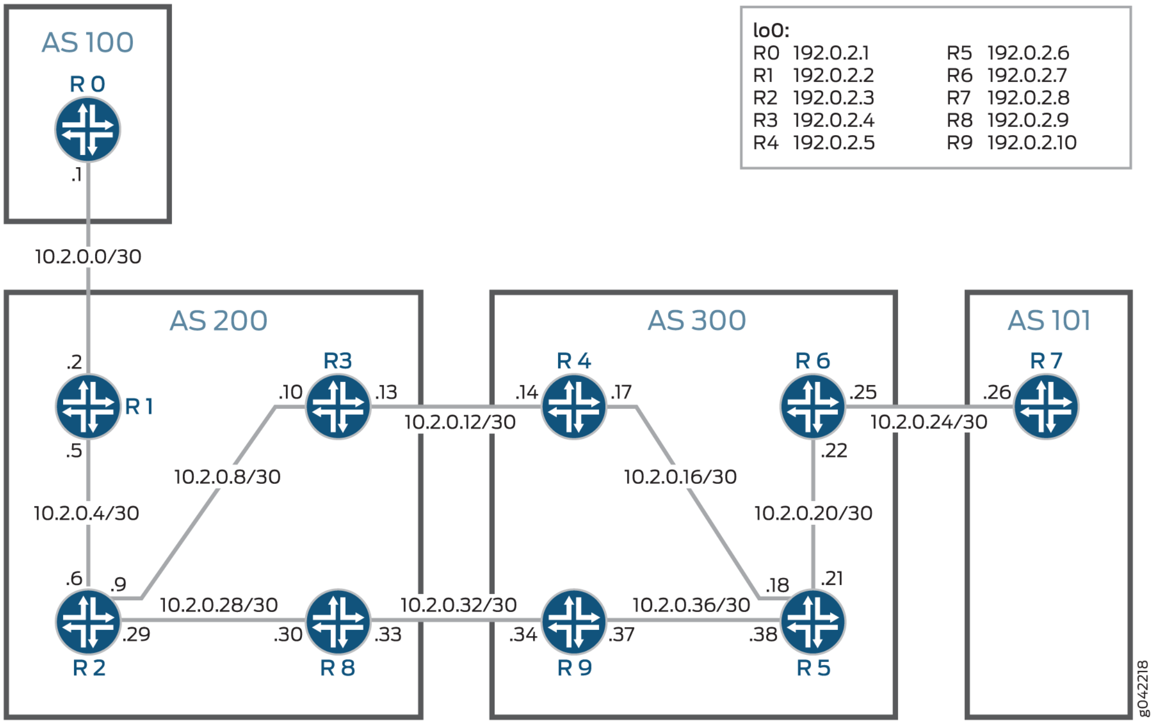
This example shows how to configure labeled-unicast egress protection in a Layer 3 VPN.

Topology

In this example, an Inter-AS Option C topology is set up by configuring two customer edge (CE) devices and six service provider edge (PE) devices in four autonomous systems. The CE devices are configured in AS100 and AS101. The PE devices are configured in AS200 and AS300.

Figure 80 on page 983 shows the topology used in this example.

Figure 80: Egress Protection in a Layer 3 VPN



lo0:

R0	192.0.2.1	R5	192.0.2.6
R1	192.0.2.2	R6	192.0.2.7
R2	192.0.2.3	R7	192.0.2.8
R3	192.0.2.4	R8	192.0.2.9
R4	192.0.2.5	R9	192.0.2.10

The aim of this example is to protect PE Router R4. Egress protection is configured on Router R4 and Router R9 so that the traffic can be routed through the backup link (R9 to R8) when Router R4 (or the link from R5 to R4) goes down. In this example, Router R4 is the protected router, Router R9 is the protector router, and Router R5 is the point of local repair (PLR).

Configuration

IN THIS SECTION

CLI Quick Configuration

To quickly configure this example, copy the following commands, paste them into a text file, remove any line breaks, change any details necessary to match your network configuration, and then copy and paste the commands into the CLI at the `[edit]` hierarchy level.

Router R0

```
set interfaces ge-0/0/0 unit 0 description toR1
set interfaces ge-0/0/0 unit 0 family inet address 10.2.0.1/30
set interfaces lo0 unit 0 family inet address 192.0.2.1/24 primary
set routing-options router-id 192.0.2.1
set routing-options autonomous-system 100
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set protocols ospf area 0.0.0.0 interface ge-0/0/0.0 metric 10
```

Router R1

```
set interfaces ge-0/0/0 unit 0 description toR0
set interfaces ge-0/0/0 unit 0 family inet address 10.2.0.2/30
set interfaces ge-0/0/0 unit 0 family mpls
set interfaces ge-0/0/1 unit 0 description toR2
set interfaces ge-0/0/1 unit 0 family inet address 10.2.0.5/30
set interfaces ge-0/0/1 unit 0 family mpls
set interfaces lo0 unit 0 family inet address 192.0.2.2/24
set routing-options router-id 192.0.2.2
set routing-options autonomous-system 200
set protocols mpls label-switched-path ToR3 to 192.0.2.4
set protocols mpls label-switched-path ToR8 to 192.0.2.9
set protocols mpls interface all
set protocols bgp group parent-vpn-peers type internal
set protocols bgp group parent-vpn-peers local-address 192.0.2.2
set protocols bgp group parent-vpn-peers family inet unicast
set protocols bgp group parent-vpn-peers family inet labeled-unicast rib inet.3
set protocols bgp group parent-vpn-peers neighbor 192.0.2.4
set protocols bgp group parent-vpn-peers neighbor 192.0.2.9
```

```
set protocols bgp group toR6 type external
set protocols bgp group toR6 multihop ttl 10
set protocols bgp group toR6 local-address 192.0.2.2
set protocols bgp group toR6 family inet-vpn unicast
set protocols bgp group toR6 peer-as 300
set protocols bgp group toR6 neighbor 192.0.2.7
set protocols ospf traffic-engineering
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set protocols ospf area 0.0.0.0 interface ge-0/0/1.0 metric 10
set protocols ldp interface ge-0/0/1.0
set protocols ldp interface lo0.0
set policy-options policy-statement child_vpn_routes term 1 from protocol bgp
set policy-options policy-statement child_vpn_routes term 1 then accept
set policy-options policy-statement child_vpn_routes term 2 then reject
set policy-options policy-statement vpnexport term 1 from protocol ospf
set policy-options policy-statement vpnexport term 1 then community add test_comm
set policy-options policy-statement vpnexport term 1 then accept
set policy-options policy-statement vpnexport term 2 then reject
set policy-options policy-statement vpnimport term 1 from protocol bgp
set policy-options policy-statement vpnimport term 1 from community test_comm
set policy-options policy-statement vpnimport term 1 then accept
set policy-options policy-statement vpnimport term 2 then reject
set policy-options community text_comm members target:1:200
set routing-instances customer-provider-vpn instance-type vrf
set routing-instances customer-provider-vpn interface ge-0/0/0.0
set routing-instances customer-provider-vpn route-distinguisher 192.0.2.4:1
set routing-instances customer-provider-vpn vrf-import vpnimport
set routing-instances customer-provider-vpn vrf-export vpnexport
set routing-instances customer-provider-vpn vrf-target target:200:1
set routing-instances customer-provider-vpn protocols ospf export child_vpn_routes
set routing-instances customer-provider-vpn protocols ospf area 0.0.0.0 interface ge-0/0/0.0
```

Router R2

```
set interfaces ge-0/0/0 unit 0 description toR3
set interfaces ge-0/0/0 unit 0 family inet address 10.2.0.9/30
set interfaces ge-0/0/0 unit 0 family mpls
set interfaces ge-0/0/1 unit 0 description toR1
set interfaces ge-0/0/1 unit 0 family inet address 10.2.0.6/30
set interfaces ge-0/0/1 unit 0 family mpls
set interfaces ge-0/0/2 unit 0 description toR8
set interfaces ge-0/0/2 unit 0 family inet address 10.2.0.29/30
```

```
set interfaces ge-0/0/2 unit 0 family mpls
set interfaces lo0 unit 0 family inet address 192.0.2.3/24
set routing-options router-id 192.0.2.3
set routing-options autonomous-system 200
set protocols mpls interface all
set protocols ospf traffic-engineering
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set protocols ospf area 0.0.0.0 interface ge-0/0/0.0 metric 10
set protocols ospf area 0.0.0.0 interface ge-0/0/1.0 metric 10
set protocols ospf area 0.0.0.0 interface ge-0/0/2.0 metric 10
set protocols ldp interface ge-0/0/0.0
set protocols ldp interface ge-0/0/1.0
set protocols ldp interface ge-0/0/2.0
set protocols ldp interface lo0.0
```

Router R3

```
set interfaces ge-0/0/0 unit 0 description toR2
set interfaces ge-0/0/0 unit 0 family inet address 10.2.0.10/30
set interfaces ge-0/0/0 unit 0 family mpls
set interfaces ge-0/0/1 unit 0 description toR4
set interfaces ge-0/0/1 unit 0 family inet address 10.2.0.13/30
set interfaces ge-0/0/1 unit 0 family mpls
set interfaces lo0 unit 0 family inet address 192.0.2.4/24
set routing-options router-id 192.0.2.4
set routing-options autonomous-system 200
set protocols mpls traffic-engineering bgp-igp-both-ribs
set protocols mpls label-switched-path ToR1 to 192.0.2.2
set protocols mpls interface all
set protocols bgp group toR4 type external
set protocols bgp group toR4 family inet unicast
set protocols bgp group toR4 family inet labeled-unicast rib inet.3
set protocols bgp group toR4 export send-pe
set protocols bgp group toR4 neighbor 10.2.0.14 peer-as 300
set protocols bgp group parent-vpn-peers type internal
set protocols bgp group parent-vpn-peers local-address 192.0.2.4
set protocols bgp group parent-vpn-peers family inet unicast
set protocols bgp group parent-vpn-peers family inet labeled-unicast rib inet.3
set protocols bgp group parent-vpn-peers export next-hop-self
set protocols bgp group parent-vpn-peers neighbor 192.0.2.2
set protocols bgp group parent-vpn-peers neighbor 192.0.2.9
set protocols ospf traffic-engineering
```

```
set protocols ospf export from-bgp
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set protocols ospf area 0.0.0.0 interface ge-0/0/0.0 metric 10
set protocols ldp interface ge-0/0/0.0
set protocols ldp interface ge-0/0/1.0
set protocols ldp interface lo0.0
set policy-options policy-statement next-hop-self term 1 then next-hop-self
set policy-options policy-statement send-pe from route-filter 192.0.2.2/24 exact
set policy-options policy-statement send-pe then accept
```

Router R4

```
set interfaces ge-0/0/0 unit 0 description toR5
set interfaces ge-0/0/0 unit 0 family inet address 10.2.0.17/30
set interfaces ge-0/0/0 unit 0 family iso
set interfaces ge-0/0/0 unit 0 family mpls
set interfaces ge-0/0/1 unit 0 description toR3
set interfaces ge-0/0/1 unit 0 family inet address 10.2.0.14/30
set interfaces ge-0/0/1 unit 0 family mpls
set interfaces lo0 unit 0 family inet address 192.0.2.5/24
set interfaces lo0 unit 0 family iso address 47.0005.80ff.f800.0000.0108.0001.0102.5507.2049.00
set routing-options router-id 192.0.2.5
set routing-options autonomous-system 300
set protocols mpls traffic-engineering bgp-igp-both-ribs
set protocols mpls label-switched-path ToR6 to 192.0.2.7
set protocols mpls interface all
set protocols mpls interface fxp.0 disable
set protocols mpls egress-protection context-identifier 203.0.113.1 primary
set protocols bgp group parent-vpn-peers type internal
set protocols bgp group parent-vpn-peers local-address 192.0.2.5
set protocols bgp group parent-vpn-peers family inet unicast
set protocols bgp group parent-vpn-peers family inet labeled-unicast rib inet.3
set protocols bgp group parent-vpn-peers family inet labeled-unicast egress-protection context-
identifier 203.0.113.1
set protocols bgp group parent-vpn-peers export next-hop-self
set protocols bgp group parent-vpn-peers neighbor 192.0.2.7
set protocols bgp group parent-vpn-peers neighbor 192.0.2.10
set protocols bgp group toR3 type external
set protocols bgp group toR3 family inet labeled-unicast rib inet.3
set protocols bgp group toR3 export send-pe
set protocols bgp group toR3 peer-as 200
set protocols bgp group toR3 neighbor 10.2.0.13
```

```
set protocols isis level 1 disable
set protocols isis level 2 wide-metrics-only
set protocols isis interface ge-0/0/0.0 level 2 metric 10
set protocols isis interface lo0.0 passive
set protocols ldp interface ge-0/0/0.0
set protocols ldp interface ge-0/0/1.0
set protocols ldp interface lo0.0
set policy-options policy-statement next-hop-self term 1 then next-hop-self
set policy-options policy-statement send-pe from route-filter 192.0.2.7/24 exact
set policy-options policy-statement send-pe then accept
```

Router R5

```
set interfaces ge-0/0/0 unit 0 description toR4
set interfaces ge-0/0/0 unit 0 family inet address 10.2.0.18/30
set interfaces ge-0/0/0 unit 0 family iso
set interfaces ge-0/0/0 unit 0 family mpls
set interfaces ge-0/0/1 unit 0 description toR6
set interfaces ge-0/0/1 unit 0 family inet address 10.2.0.21/30
set interfaces ge-0/0/1 unit 0 family iso
set interfaces ge-0/0/1 unit 0 family mpls
set interfaces ge-0/0/2 unit 0 description toR9
set interfaces ge-0/0/2 unit 0 family inet address 10.2.0.38/30
set interfaces ge-0/0/2 unit 0 family iso
set interfaces ge-0/0/2 unit 0 family mpls
set interfaces lo0 unit 0 family inet address 192.0.2.6/24
set interfaces lo0 unit 0 family iso address 47.0005.80ff.f800.0000.0108.0001.0102.5507.2050.00
set routing-options router-id 192.0.2.6
set routing-options autonomous-system 300
set protocols mpls interface all
set protocols mpls interface fxp0.0 disable
set protocols isis backup-spf-options per-prefix-calculation
set protocols isis level 1 disable
set protocols isis level 2 wide-metrics-only
set protocols isis interface all node-link-protection
set protocols isis interface fxp0.0 disable
set protocols isis interface ge-0/0/0.0 link-protection
set protocols isis interface ge-0/0/0.0 level 2 metric 10
set protocols isis interface ge-0/0/1.0 link-protection
set protocols isis interface ge-0/0/1.0 level 2 metric 10
set protocols isis interface ge-0/0/2.0 link-protection
set protocols isis interface ge-0/0/2.0 level 2 metric 10
```

```
set protocols isis interface lo0.0 passive
set protocols ldp track-igp-metric
set protocols ldp interface all
set protocols ldp interface fxp0.0 disable
```

Router R6

```
set interfaces ge-0/0/0 unit 0 description toR7
set interfaces ge-0/0/0 unit 0 family inet address 10.2.0.25/30
set interfaces ge-0/0/0 unit 0 family iso
set interfaces ge-0/0/0 unit 0 family mpls
set interfaces ge-0/0/1 unit 0 description toR5
set interfaces ge-0/0/1 unit 0 family inet address 10.2.0.22/30
set interfaces ge-0/0/1 unit 0 family iso
set interfaces ge-0/0/1 unit 0 family mpls
set interfaces lo0 unit 0 family inet address 192.0.2.7/24
set interfaces lo0 unit 0 family iso address 47.0005.80ff.f800.0000.0108.0001.0102.5507.2048.00
set routing-options router-id 192.0.2.7
set routing-options autonomous-system 300
set protocols mpls label-switched-path ToR4 to 192.0.2.5
set protocols mpls label-switched-path ToR9 to 192.0.2.10
set protocols mpls interface all
set protocols bgp group parent-vpn-peers type internal
set protocols bgp group parent-vpn-peers local-address 192.0.2.7
set protocols bgp group parent-vpn-peers family inet unicast
set protocols bgp group parent-vpn-peers family inet labeled-unicast rib inet.3
set protocols bgp group parent-vpn-peers neighbor 192.0.2.5
set protocols bgp group parent-vpn-peers neighbor 192.0.2.10
set protocols bgp group toR1 type external
set protocols bgp group toR1 multihop ttl 10
set protocols bgp group toR1 local-address 192.0.2.7
set protocols bgp group toR1 family inet-vpn unicast
set protocols bgp group toR1 peers-as 200
set protocols bgp group toR1 neighbor 192.0.2.2
set protocols isis level 1 disable
set protocols isis level 2 wide-metrics-only
set protocols isis interface ge-0/0/1.0 level 2 metric 10
set protocols isis interface lo0.0 passive
set protocols ldp interface ge-0/0/1.0
set protocols ldp interface lo0.0
set policy-options policy-statement child-vpn-routes term 1 from protocol bgp
set policy-options policy-statement child-vpn-routes term 1 then accept
```

```
set policy-options policy-statement child-vpn-routes term 2 then reject
set policy-options policy-statement vpnexport term 1 from protocol ospf
set policy-options policy-statement vpnexport term 1 then community add test_comm
set policy-options policy-statement vpnexport term 1 then accept
set policy-options policy-statement vpnexport term 2 then reject
set policy-options policy-statement vpnimport term 1 from protocol bgp
set policy-options policy-statement vpnimport term 1 from community test_comm
set policy-options policy-statement vpnimport term 1 then accept
set policy-options policy-statement vpnimport term 2 then reject
set policy-options community test_comm members target:1:300
set routing-instances customer-provider-vpn instance-type vrf
set routing-instances customer-provider-vpn interface ge-0/0/0.0
set routing-instances customer-provider-vpn route-distinguisher 192.0.2.5:1
set routing-instances customer-provider-vpn vrf-import vpnimport
set routing-instances customer-provider-vpn vrf-export vpnexport
set routing-instances customer-provider-vpn vrf-target target:300:1
set routing-instances customer-provider-vpn protocols ospf export child-vpn-routes
set routing-instances customer-provider-vpn protocols ospf area 0.0.0.0 interface ge-0/0/0.0
```

Router R7

```
set interfaces ge-0/0/0 unit 0 description toR6
set interfaces ge-0/0/0 unit 0 family inet address 10.2.0.26/30
set interfaces ge-0/0/0 unit 0 family mpls
set interfaces lo0 unit 0 family inet address 192.0.2.8/24 primary
set routing-options router-id 192.0.2.8
set routing-options autonomous-system 101
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set protocols ospf area 0.0.0.0 interface ge-0/0/0.0 metric 10
```

Router R8

```
set interfaces ge-0/0/0 unit 0 description toR9
set interfaces ge-0/0/0 unit 0 family inet address 10.2.0.33/30
set interfaces ge-0/0/0 unit 0 family mpls
set interfaces ge-0/0/1 unit 0 description toR2
set interfaces ge-0/0/1 unit 0 family inet address 10.2.0.30/30
set interfaces ge-0/0/1 unit 0 family mpls
set interfaces lo0 unit 0 family inet address 192.0.2.9/24
set routing-options router-id 192.0.2.9
set routing-options autonomous-system 200
```

```
set protocols mpls traffic-engineering bgp-igp-both-ribs
set protocols mpls label-switched-path ToR1 to 192.0.2.2
set protocols mpls interface all
set protocols bgp group toR9 type external
set protocols bgp group toR9 family inet unicast
set protocols bgp group toR9 family inet labeled-unicast rib inet.3
set protocols bgp group toR9 export send-pe
set protocols bgp group toR9 neighbor 10.2.0.34 peer-as 300
set protocols bgp group parent-vpn-peers type internal
set protocols bgp group parent-vpn-peers local-address 192.0.2.9
set protocols bgp group parent-vpn-peers family inet unicast
set protocols bgp group parent-vpn-peers family inet labeled-unicast rib inet.3
set protocols bgp group parent-vpn-peers export next-hop-self
set protocols bgp group parent-vpn-peers neighbor 192.0.2.2
set protocols bgp group parent-vpn-peers neighbor 192.0.2.4
set protocols ospf traffic-engineering
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set protocols ospf area 0.0.0.0 interface ge-0/0/1.0 metric 10
set protocols ldp interface ge-0/0/0.0
set protocols ldp interface ge-0/0/1.0
set protocols ldp interface lo0.0
set policy-options policy-statement from-bgp from protocol bgp
set policy-options policy-statement from-bgp then metric add 100
set policy-options policy-statement from-bgp then accept
set policy-options policy-statement next-hop-self term 1 then next-hop-self
set policy-options policy-statement send-pe from route-filter 192.0.2.2/24 exact
set policy-options policy-statement send-pe then accept
```

Router R9

```
set interfaces ge-0/0/0 unit 0 description toR8
set interfaces ge-0/0/0 unit 0 family inet address 10.2.0.34/30
set interfaces ge-0/0/0 unit 0 family mpls
set interfaces ge-0/0/1 unit 0 description toR5
set interfaces ge-0/0/1 unit 0 family inet address 10.2.0.37/30
set interfaces ge-0/0/1 unit 0 family iso
set interfaces ge-0/0/1 unit 0 family mpls
set interfaces lo0 unit 0 family inet address 192.0.2.10/24
set interfaces lo0 unit 0 family iso address 47.0005.80ff.f800.0000.0108.0001.0102.5507.2062.00
set routing-options router-id 192.0.2.10
set routing-options autonomous-system 300
set protocols mpls traffic-engineering bgp-igp-both-ribs
```

```
set protocols mpls label-switched-path ToR6 to 192.0.2.7
set protocols mpls interface all
set protocols mpls egress-protection context-identifier 203.0.113.1 protector
set protocols bgp group parent-vpn-peers type internal
set protocols bgp group parent-vpn-peers local-address 192.0.2.10
set protocols bgp group parent-vpn-peers family inet unicast
set protocols bgp group parent-vpn-peers family inet labeled-unicast rib inet.3
set protocols bgp group parent-vpn-peers family inet labeled-unicast egress-protection
set protocols bgp group parent-vpn-peers export next-hop-self
set protocols bgp group parent-vpn-peers neighbor 192.0.2.7
set protocols bgp group parent-vpn-peers neighbor 192.0.2.5
set protocols bgp group toR8 type external
set protocols bgp group toR8 family inet labeled-unicast rib inet.3
set protocols bgp group toR8 export send-pe
set protocols bgp group toR8 neighbor 10.2.0.33 peer-as 200
set protocols isis level 1 disable
set protocols isis level 2 wide-metrics-only
set protocols isis interface ge-0/0/1.0 level 2 metric 10
set protocols isis interface lo0.0 passive
set protocols ldp interface ge-0/0/0.0
set protocols ldp interface ge-0/0/1.0
set protocols ldp interface lo0.0
set policy-options policy-statement next-hop-self term 1 then next-hop-self
set policy-options policy-statement send-pe from route-filter 192.0.2.7/24 exact
set policy-options policy-statement send-pe then accept
```

Configuring Egress Protection in Layer 3 VPNs

Step-by-Step Procedure

The following example requires that you navigate various levels in the configuration hierarchy. For information about navigating the CLI, see *Using the CLI Editor in Configuration Mode* in the CLI User Guide.

To configure labeled unicast egress protection:

1. Configure the interfaces on each router, for example:

```
[edit interfaces]
user@R4# set ge-0/0/0 unit 0 description toR5
user@R4# set ge-0/0/0 unit 0 family inet address 10.2.0.17/30
```

```
user@R4# set ge-0/0/0 unit 0 family iso
user@R4# set ge-0/0/0 unit 0 family mpls
```

```
user@R4# set ge-0/0/1 unit 0 description toR3
user@R4# set ge-0/0/1 unit 0 family inet address 10.2.0.14/30
user@R4# set ge-0/0/1 unit 0 family mpls
```

```
user@R4# set lo0 unit 0 family inet address 192.0.2.5/24
user@R4# set lo0 unit 0 family iso address 47.0005.80ff.f800.0000.0108.0001.0102.5507.2049.00
```

2. Configure the router ID and autonomous system (AS) number for each router, for example:

```
[edit routing-options]
user@R4# set router-id 192.0.2.5
user@R4# set autonomous-system 300
```

   In this example, the router ID is chosen to be identical to the loopback address configured on the router.

3. Configure the protocols on each router, for example:

```
[edit protocols]
user@R4# set mpls traffic-engineering bgp-igp-both-ribs
user@R4# set mpls label-switched-path ToR6 to 192.0.2.7
user@R4# set mpls interface all
user@R4# set mpls interface fxp.0 disable
user@R4# set bgp group parent-vpn-peers type internal
user@R4# set bgp group parent-vpn-peers local-address 192.0.2.5
user@R4# set bgp group parent-vpn-peers family inet unicast
user@R4# set bgp group parent-vpn-peers family inet labeled-unicast rib inet.3
user@R4# set bgp group parent-vpn-peers export next-hop-self
user@R4# set bgp group parent-vpn-peers neighbor 192.0.2.7
user@R4# set bgp group parent-vpn-peers neighbor 192.0.2.10
user@R4# set bgp group toR3 type external
user@R4# set bgp group toR3 family inet labeled-unicast rib inet.3
user@R4# set bgp group toR3 export send-pe
user@R4# set bgp group toR3 peer-as 200
user@R4# set bgp group toR3 neighbor 10.2.0.13
user@R4# set isis level 1 disable
```

```
user@R4# set isis level 2 wide-metrics-only
user@R4# set isis interface ge-0/0/0.0 level 2 metric 10
user@R4# set isis interface lo0.0 passive
user@R4# set ldp interface ge-0/0/0.0
user@R4# set ldp interface ge-0/0/1.0
user@R4# set ldp interface lo0.0
```

4. Configure routing policies on all PE routers and AS border routers (Routers R1, R3, R4, R6, R8, and R9), for example:

```
user@R4# set policy-options policy-statement next-hop-self term 1 then next-hop-self
user@R4# set policy-options policy-statement send-pe from route-filter 192.0.2.7/24 exact
user@R4# set policy-options policy-statement send-pe then accept
```

5. Configure the VPN routing instance on Routers R1 and R6.

```
user@R1# set routing-instances customer-provider-vpn instance-type vrf
user@R1# set routing-instances customer-provider-vpn interface ge-0/0/0.0
user@R1# set routing-instances customer-provider-vpn route-distinguisher 192.0.2.4:1
user@R1# set routing-instances customer-provider-vpn vrf-import vpnimport
user@R1# set routing-instances customer-provider-vpn vrf-export vpnexport
user@R1# set routing-instances customer-provider-vpn vrf-target target:200:1
user@R1# set routing-instances customer-provider-vpn protocols ospf export child_vpn_routes
user@R1# set routing-instances customer-provider-vpn protocols ospf area 0.0.0.0 interface
ge-0/0/0.0
```

and

```
user@R6# set routing-instances customer-provider-vpn instance-type vrf
user@R6# set routing-instances customer-provider-vpn interface ge-0/0/0.0
user@R6# set routing-instances customer-provider-vpn route-distinguisher 192.0.2.5:1
user@R6# set routing-instances customer-provider-vpn vrf-import vpnimport
user@R6# set routing-instances customer-provider-vpn vrf-export vpnexport
user@R6# set routing-instances customer-provider-vpn vrf-target target:300:1
user@R6# set routing-instances customer-provider-vpn protocols ospf export child-vpn-routes
user@R6# set routing-instances customer-provider-vpn protocols ospf area 0.0.0.0 interface
ge-0/0/0.0
```

6. Configure egress protection for Router R4, setting Router R4 as the protected router and Router R9 as the protector.

   ```
   user@R4# set protocols mpls egress-protection context-identifier 203.0.113.1 primary
   user@R4# set protocols bgp group parent-vpn-peers family inet labeled-unicast egress-
   protection context-identifier 203.0.113.1
   ```

   and

   ```
   user@R9# set protocols mpls egress-protection context-identifier 203.0.113.1 protector
   user@R9# set protocols bgp group parent-vpn-peers family inet labeled-unicast egress-
   protection
   ```

Results

From configuration mode, confirm your configuration by entering the `show interfaces`, `show routing-options`, `show protocols`, `show policy-options` (if applicable), and `show routing-instances` (if applicable) commands.

If the output does not display the intended configuration, repeat the instructions in this example to correct the configuration.

```
user@R4# show interfaces
ge-0/0/0 {
    unit 0 {
        description toR5;
        family inet {
            address 10.2.0.17/30;
        }
        family iso;
        family mpls;
    }
}
ge-0/0/1 {
    unit 0 {
        description toR3;
        family inet {
            address 10.2.0.14/30;
        }
        family mpls;
    }
```

```
}
lo0 {
    unit 0 {
        family inet {
            address 192.0.2.5/24;
        }
        family iso {
            address 47.0005.80ff.f800.0000.0108.0001.0102.5507.2049.00;
        }
    }
}
```

```
user@R4# show routing-options
router-id 192.0.2.5;
autonomous-system 300;
```

```
user@R4# show protocols
mpls {
    traffic-engineering bgp-igp-both-ribs;
    label-switched-path ToR6 {
        to 192.0.2.7;
    }
    interface all;
    interface fxp0.0 {
        disable;
    }
    egress-protection {
        context-identifier 203.0.113.1 {
            primary;
        }
    }
}
bgp {
    group parent-vpn-peers {
        type internal;
        local-address 192.0.2.5;
        family inet {
            unicast;
            labeled-unicast {
                rib {
```

```
                    inet.3;
                }
                egress-protection {
                    context-identifier {
                        203.0.113.1;
                    }
                }
            }
        }
        export next-hop-self;
        neighbor 192.0.2.7;
        neighbor 192.0.2.10;
    }
    group toR3 {
        type external;
        family inet {
            unicast;
            labeled-unicast {
                rib {
                    inet.3;
                }
            }
        }
        export send-pe;
        peer-as 200;
        neighbor 10.2.0.13;
    }
}
isis {
    level 1 disable;
    level 2 wide-metrics-only;
    interface ge-0/0/0.0 {
        level 2 metric 10;
    }
    interface lo0.0 {
        passive;
    }
}
ldp {
    interface ge-0/0/0.0;
    interface ge-0/0/1.0;
```

```
    interface lo0.0;
}
```

```
user@R4# show policy-options
policy-statement next-hop-self {
    term 1 {
        then {
            next-hop self;
        }
    }
}
policy-statement send-pe {
    from {
        route-filter 192.0.2.7/24 exact;
    }
    then accept;
}
```

If you are done configuring the router, enter `commit` from configuration mode.

Repeat the procedure for every router in this example, using the appropriate interface names and addresses for each router.

Verification

IN THIS SECTION

Verifying That Egress Protection Is Enabled

Purpose

Verify that egress protection is enabled on the protected router, Router R4.

Action

Run **show bgp neighbor** on Router R4 to verify that egress protection is enabled.

```
user@R4> show bgp neighbor
Peer: 192.0.2.10+45824 AS 300    Local: 192.0.2.5+27630 AS 300
  Type: Internal    State: Established    Flags: <Sync>
  Last State: OpenConfirm   Last Event: RecvKeepAlive
  Last Error: None
  Export: [ next-hop-self ]
  Options: <Preference LocalAddress AddressFamily Refresh>
  Address families configured: inet-unicast inet-labeled-unicast
  Local Address: 192.0.2.5 Holdtime: 90 Preference: 170
  NLRI configured with egress-protection: inet-labeled-unicast
  Egress-protection NLRI inet-labeled-unicast context-identifier: 203.0.113.1
  Number of flaps: 0
  ...
```

Verifying the State of the Protected ASBR as ’primary’

Purpose

Verify that the state of the protected AS border router, Router R4, is ’primary’.

Action

Run **show mpls context-identifier** on Router R4.

```
user@R4> show mpls context-identifier
ID                 Type        Metric     ContextTable
203.0.113.1          primary     1
Total 1, Primary 1, Protector 0
```

Verifying the State of the Protector ASBR as ’protector’

Purpose

Verify that the state of the protector AS border router, Router R9, is ’protector’.

Action

Run **show mpls context-identifier** on Router R9.

```
user@R9> show mpls context-identifier
ID                 Type        Metric     ContextTable
203.0.113.1          protector    16777215    __203.0.113.1__.mpls.0
Total 1, Primary 0, Protector 1
```

SEE ALSO

egress-protection (BGP)

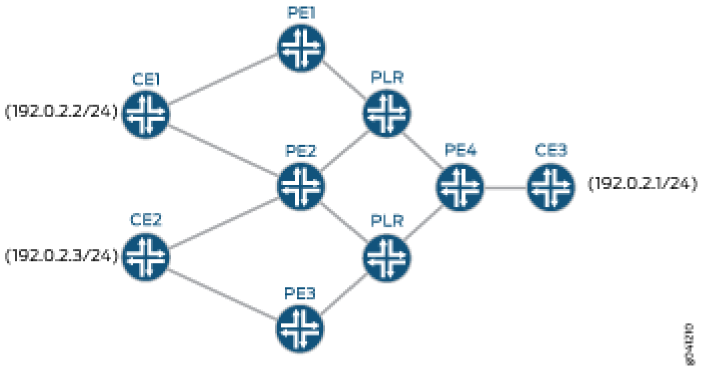
Egress Protection for Layer 3 VPN Edge Protection Overview

IN THIS SECTION

Typically, Layer 3 VPN service restoration for multihomed customer edge (CE) routers depends on the ingress provider edge (PE) router to detect the egress PE link or node failure and switch traffic to the backup PE router. To achieve faster restoration, a protector mechanism for the PE router can be used to perform local restoration of the service immediately in case of an egress PE node failure. This mechanism requires the router at the point of local repair (PLR) to redirect VPN traffic to a protector PE router for fast reroute of traffic.

The following topology describes the concept of egress protection.

Figure 81: Sample Topology for Egress Protection



In this topology:

Router PE3 acts as the protector for the PE2 Layer 3 VPN routing instances or subnets.

The CE routers are part of a VPN where Router CE1 is multihomed with Router PE1 and Router PE2. Likewise, Router CE2 is multihomed with Routers PE2 and PE3.

Router PE1 can be the originator for the context identifier for Router CE1, while Router PE2 is the protector for that context identifier. Likewise, PE2 can be the originator for the context identifier for Router CE2, while Router PE3 is the protector for that context identifier.

The working path taken by Router PE4 might be through PLR>PE2 for both Router CE1 and Router CE2. The backup path for Router CE1 is through PLR>PE1. The backup path for Router CE2 is through PLR>PE3. Traffic flows through the working path under normal circumstances.

When Router PE4 detects a PE2 node or link failure, traffic is rerouted from the working path to the protected path. In the normal failover process, the detection of failure and the recovery rely on the control plane and is therefore relatively slow.

Typically, if there is a link or node failure in the core network, the egress PE router would have to rely on the ingress PE router to detect the failure and switch over to the backup path, because a local repair option for egress failure is not available.

To provide a local repair solution for the egress PE link or node failure, a mechanism known as egress protection can be used to repair and restore the connection quickly. If egress protection is configured, the PLR router detects the PE2 link or node failure and reroutes traffic through the protector Router PE3 using the backup LDP-signaled label-switched path (LSP). The PLR router uses per-prefix loop-free alternate routes to program the backup next hop through Router PE3, and traffic is forwarded to Routers CE1 and CE2 using the alternate paths. This restoration is done quickly after the PLR router detects the Router PE2 egress node or link failure.

The dual protection mechanism can also be used for egress protection where the two PE routers can simultaneously act as the primary PE router and the protector PE router for their respective context ID routes or next hops.

Router Functions

In Figure 81 on page 1001, the following routers perform the following functions:

Protected PE Router

The protected PE, PE2, performs the following functions:

- Updates a context identifier for the BGP next hop for the Layer 3 VPN prefix.
- Advertises the context identifier to the IS-IS domain.

Protector PE Router

The protector PE router, PE3, performs the following functions:

- Advertises the context identifier to the IS-IS domain with a high metric. The high IGP metric (configurable) along with the LDP label ensures that the PLR router uses the LDP-signaled backup LSP in the event of an egress PE router failure.
- Builds a context-label table for route lookup and a backup forwarding table for the protected PE router (PE2).

NOTE: The protector PE router should not be in the forwarding path to the primary PE router.

PLR Router

The router acting as the point of local repair (PLR) performs the following functions:

- Computes per-prefix loop-free alternate routes. For this computation to work, the configuration of the `node-link-protection` statement and the `backup-spf-options per-prefix-calculation` statement is necessary at the `[edit protocols isis]` hierarchy level.
- Installs backup next hops for the context identifier through the PE3 router (protector PE).
- Detects PE router failure and redirects the transport LSP traffic to the protector.

NOTE: The PLR router must be directly connected to the protector router (in this case, PE3). If not, the loop-free alternate route cannot find the backup path to the protector. This limitation is removed in Junos OS Release 13.3 and later.

Protector and Protection Models

Protector is a new role or function for the restoration of egress PE node failure. This role could be played by a backup egress PE router or any other node that participates in the VPN control plane for VPN prefixes that require egress node protection. There are two protection models based on the location and role of a protector:

- Co-located protector—In this model, the protector PE router and the backup PE router configurations are done on the same router. The protector is co-located with the backup PE router for the protected prefix, and it has a direct connection to the multihomed site that originates the protected prefix. In the event of an egress PE failure, the protector receives traffic from the PLR router and routes the traffic to the multihomed site.
- Centralized protector—In this model, the protector PE router and the backup PE router are different. The centralized protector might not have a direct connection to the multihomed site. In the event of an egress PE link or node failure, the centralized protector reroutes the traffic to the backup egress PE router with the VPN label advertised for the backup egress PE router that takes over the role of sending traffic to the multihomed site.

A network can use either of the protection models or a combination of both, depending on the requirement.

As a special scenario of egress node protection, if a router is both a Protector and a PLR, it installs backup next hops to protect the transport LSP. In particular, it does not need a bypass LSP for local repair.

In the Co-located protector model, the PLR or the Protector is directly connected to the CE via a backup AC, while in the Centralized protector model, the PLR or the protector has an MPLS tunnel to the backup PE. In either case, the PLR or the Protector will install a backup next hop with a label followed by a lookup in a `context label` table, i.e. `__context__.mpls.0`. When the egress node fails, the PLR or the Protector will switch traffic to this backup next hop in PFE. The outer label (th etransport LSP label) of packets is popped, and the inner label (the layer 3 VPN label allocated by the egress node) is looked up in `__context__.mpls.0`, which results in forwarding the packets directly to the CE (in Collocated protector model) or the backup PE (in Centralized protector model).

For more information about egress PE failure protection, see Internet draft draft-minto-2547-egress-node-fast-protection-00, *2547 egress PE Fast Failure Protection.*.

IGP Advertisement Model

Egress protection availability is advertised in the interior gateway protocol (IGP). Label protocols along with Constrained Shortest Path First (CSPF) use this information to do egress protection.

For Layer 3 VPNs, the IGP advertisements can be of the following types:

- Context identifier as a stub link (supported in Junos OS 11.4 R3 and later). A link connecting a stub node to a transit node is a stub link.
- Context identifier as a stub alias node (supported in Junos OS 13.3 and later).
- Context identifier as a stub proxy node (supported in Junos OS 13.3 and later).

By default, the stub link is used. To enable enhanced point-of-local-repair (PLR) functionality, in which the PLR reroutes service traffic during an egress failure, configure a stub alias node or a stub proxy node as follows:

```
[edit protocols mpls egress-protection context-identifier 192.0.2.6]
user@host# set advertise-mode ?
Possible completions:
  stub-alias           Alias
  stub-proxy           Proxy
```

The two methods offer different advantages, depending on the needs of your network deployment.

Context Identifier as a Stub Alias Node

In the stub alias method, the LSP end-point address has an explicit backup egress node where the backup can be learned or configured on the penultimate hop node of a protected LSP. With this model, the penultimate hop node of a protected LSP sets up the bypass LSP tunnel to back up the egress node by avoiding the primary egress node. This model requires a Junos OS upgrade in core nodes, but is flexible enough to support all traffic engineering constraints.

The PLR learns that the context ID has a protector. When the primary context ID goes down, packets are rerouted to the protector by way of a pre-programmed backup path. The context ID and protector mapping are configured or learned on the PLR and signaled in the IGP from the protector. A routing table called inet.5 on the PLR provides the configured or IGP-learned details.

IS-IS advertises context IDs into the TED through an IP address TLV. IS-IS imports this TLV into the TED as extended information. IS-IS advertises the protector TLV routes in the inet.5 route for the context ID with protocol next hop being the protector’s router ID. If the protector TLV has a label, the label is added to the route in the inet.5 routing table for LDP to use.

CSPF considers the IP address TLV for tunnel endpoint computation.

With the stub alias model, the protector LSP setup does not require any changes in any nodes. But bypass LSP setup for node protection requires changes in the PHN and the protector router.

When RSVP sets up bypass for node protection LSP, RSVP also performs a lookup for the protector if the PLR is the penultimate hop of the LSP. If the protector is available for the LSP destination, it uses CSPF to compute a path with a constraint that excludes the egress PE and sets up a bypass LSP destination to the context ID if one is not already set up. When setting up a bypass LSP to the context ID, the PLR unsets all protection options.

LDP is useful in the case when the network supports 100 percent LFA coverage but does not support 100 percent per-prefix LFA coverage. LDP sets up a backup path with the protector with the context label advertised by the protector to the service point.

In networks in which 100 percent LFA coverage is not available, it is useful to have backup LSP LFAs with RSVP-based tunnels.

In a steady state, the forwarding is the same as on any other protected LSP in the PLR. In the protector, the non-null label that is advertised and signaled for the context ID has the table next hop point to the MPLS context table, where the peers' labels are programmed.

During a failure, the PLR swaps the transport label with the bypass LSP for the context ID or swaps the label context-label (the protector-advertised label for the context ID) and pushes the transport label to the protector lo0 interface address.

Context Identifier as a Stub Proxy Node

Context identifier as a stub proxy node (supported in Junos OS 13.3 and later). A stub node is one that only appears at the end of an AS path, which means it does not provide transit service. In this mode, known as the virtual or proxy mode, the LSP end-point address is represented as a node with bidirectional links, with the LSP's primary egress node and backup egress node. With this representation, the penultimate hop of the LSP primary egress point can behave like a PLR in setting up a bypass tunnel to back up the egress by avoiding the primary egress node. This model has the advantage that you do not need to upgrade Junos OS on core nodes and will thereby help operators to deploy this technology.

The context ID is represented as a node in the traffic engineering (TE) and IGP databases. The primary PE device advertises the context node into the IGP and TE databases. The primary PE device and the protected PE device support one link to the context node with a bandwidth and a TE metric. Other TE characteristics of TE links are not advertised by Junos OS.

In IS-IS, the primary PE router advertises the proxy node along with links to the primary router and the protector router. The primary and the protector routers advertise links to the proxy node. The proxy node builds the following information.

- System ID—Binary-coded decimal based on the context ID.
- Host name—Protector-name:context ID
- LSP-ID—<System-ID>.00
- PDU type—Level 2 and Level 1, based on the configuration
- LSP attributes:

- Overload—1
- IS_TYPE_L1(0x01) | IS_TYPE_L2(0x02) for the level 2 PDU
- IS_TYPE_L1 for level 1
- Multiarea—No
- All other attributes—0

The proxy node only contains area, MT, host name, router ID, protocols and IS reachabilty TLVs. The area, MT, authentication, and protocols TLV are the same as on the primary. The IS reachability TLVs contains two links called Cnode-primary-link and Cnode-protector-link. Both links include TE TLVs. The following TE-link-TLVs are advertised in context links:

- IPv4 interface or neighbor address
- Maximum bandwidth
- TE default metric
- Link (local or remote) Identifiers

Sub TLV values:

- Bandwidth—zero
- TE metric—Maximum TE metric
- Interface address—context ID
- Protector neighbor address—protector router ID
- Primary neighbor address—protected router ID
- Link local-ID protector—0x80fffff1
- Link local-ID primary—0x80fffff2
- Link remote-ID protector—Learned from protector
- Link remote-ID primary—Learned from primary

Protected PE links to context node (primary advertises the link with the following details):

- Bandwidth—Maximum
- TE metric—1
- Interface address—Router ID

- Context neighbor address—Context ID
- Link local-ID to context node—Automatically generated (similar to a sham link)
- Link remote-ID to context node—0x80fffff2

Protector PE links to context node:

- The protector advertises unnumbered transit links with the maximum routable link metric and the maximum TE metric and zero bandwidth to the context node. Other TE characteristics are not advertised.

Unnumbered links are advertised with the following attributes:

- bandwidth—0
- TE metric—MAX TE metric
- Interface address—Router ID
- Context neighbor address—Context ID
- Link local ID to context node—Autogenerated (similar to a sham link)
- Link remote ID to context node—0x80fffff1

In RSVP, the behavior changes are only in the protector and primary routers. RSVP terminates the LSP and the bypass LSP to the context ID. If the context ID is the protector, a non-null label is signaled. Otherwise, it will be based on the configuration or the requested label type. RSVP verifies the Explicit Route Object (ERO) from the path for itself and the context ID. RSVP sends the Resv message with two Record Route Object (RRO) objects—one for the context ID and one for itself. This simulates the penultimate-hop node (PHN) to do node protection with the protector for the primary for context ID LSP. As the fast reroute (FRR)-required bypass, the LSP has to merge back to the protector LSP PHN setup bypass to context ID through the protector by avoiding the primary.

The protector also terminates the backup LSP for the context ID to keep the protected LSP alive during a failure until the ingress node resignals the LSP. The new LSP is reestablished through the protector, but this LSP is not used for service traffic as service protocol does not use the context ID. The LSP traverses through the protector even if the primary comes up. Only reoptimization resignals the LSP through the primary. In stub proxy mode, the bypass LSP with constraints is not supported.

LDP cannot use the stub proxy method due to the inflated metric advertised in the IGP.

With regard the forwarding state, a PE router that protects one or more segments that are connected to another PE is referred to as a protector PE. A protector PE must learn the forwarding state of the segments that it is protecting from the primary PE that is being protected.

For a given segment, if the protector PE is not directly connected to the CE device associated with the segment, it must also learn the forwarding state from at least one backup PE. This situation might arise only in the case of egress PE failure protection.

A protector PE maintains forwarding state for a given segment in the context of the primary PE. A protector PE might maintain state for only a subset of the segments on the primary PE or for all the segments on the primary PE.

Example: Configuring MPLS Egress Protection for Layer 3 VPN Services

This example describes a local repair mechanism for protecting Layer 3 VPN services against egress provider edge (PE) router failure in a scenario where the customer edge (CE) routers are multihomed with more than one PE router.

The following terminology is used in this example:

- Originator PE router—A PE router with protected routing instances or subnets that distributes the primary Layer 3 VPN router.
- Backup PE router—A PE router that announces a backup Layer 3 VPN route.
- Protector PE router—A router that cross-connects VPN labels distributed by the originator PE router to the labels originated by the backup PE router. The protector PE router can also be a backup PE router.
- Transport LSP—An LDP-signaled label-switched path (LSP) for BGP next hops.
- PLR—A router acting as the point of local repair (PLR) that can redirect Layer 3 VPN traffic to a protector PE router to enable fast restoration and reroute.
- Loop-free alternate routes—A technology that essentially adds IP fast-reroute capability for the interior gateway protocol (IGP) by precomputing backup routes for all the primary routes of the IGP. In the context of this document, the IGP is IS-IS.
- Multihoming—A technology that enables you to connect a CE device to multiple PE routers. In the event that a connection to the primary PE router fails, traffic can be automatically switched to the backup PE router.
- Context identifier—An IPv4 address used to identify the VPN prefix that requires protection. The identifier is propagated to the PE and PLR core routers, making it possible for the protected egress PE router to signal the egress protection to the protector PE router.
- Dual protection—A protection mechanism where two PE routers can simultaneously act as the primary PE router and the protector PE router for their respective context ID routes or next hops. For example, between the two PE routers PE1 and PE2, PE1 could be a primary PE router for context identifier 203.0.113.1 and protector for context identifier 203.0.113.2 Likewise, the PE2 router could

be a protector for context identifier 203.0.113.1 and a primary PE router for context identifier 203.0.113.2.

Example: Configuring Egress Protection for Layer 3 VPN Services

IN THIS SECTION

This example shows how to configure egress protection for fast restoration of Layer 3 VPN services.

Requirements

This example uses the following hardware and software components

- MX Series 5G Universal Routing Platforms
- Tunnel PICs or the configuration of the Enhanced IP Network Services mode (using the `network-services enhanced-ip` statement at the `[edit chassis]` hierarchy level).
- Junos OS Release 11.4R3 or later running on the devices

Before you begin:

- Configure the device interfaces. See the *Junos OS Network Interfaces Configuration Guide*.
- Configure the following routing protocols on all the PE and PLR routers.
  - MPLS, LSPs, and LDP. See the *Junos OS MPLS Applications Configuration Guide*.
  - BGP and IS-IS. See the *Junos OS Routing Protocols Configuration Guide*.
- Configure Layer 3 VPNs. See the *Junos OS VPNs Configuration Guide*.

Overview

Typically, Layer 3 VPN service restoration, in case of egress PE router failure (for multihomed customer edge [CE] routers), depends on the ingress PE router to detect the egress PE node failure and switch traffic to the backup PE router for multihomed CE sites.

Junos OS Release 11.4R3 or later enables you to configure egress protection for Layer 3 VPN services that protects the services from egress PE node failure in a scenario where the CE site is multihomed with more than one PE router. The mechanism enables local repair to be performed immediately upon an egress node failure. The router acting as the point of local repair (PLR) redirects VPN traffic to a protector PE router for restoring service quickly, achieving fast protection that is comparable to MPLS fast reroute.

The statements used to configure egress protection are:

- `egress-protection`—When configured at the [edit protocols mpls] hierarchy level, this statement specifies protector information and the context identifier for the Layer 3 VPN and edge protection virtual circuit:

  ```
  [edit protocols mpls]
  egress-protection {
      context-identifier context-id {
          primary | protector;
          metric igp-metric-value;
      }
  }
  ```

  When configured at the `[edit protocols bgp group group-name family inet-vpn unicast]`, `[edit protocols bgp group group-name family inet6-vpn unicast]`, or `[edit protocols bgp group group-name family iso-vpn unicast]` hierarchy levels, the egress-protection statement specifies the context identifier that enables egress protection for the configured BGP VPN network layer reachability information (NLRI).

  ```
  [edit protocols bgp]
  group internal {
      type internal;
  local-address ip-address;
      family <inet-vpn|inet6-vpn|iso-vpn> {
          unicast {
              egress-protection {
                  context-identifier {
                      context-id-ip-address;
                      }
                  }
              }
          }
      }
  ```

When configured at the `[edit routing-instances]` hierarchy level, the `egress-protection` statement holds the context identifier of the protected PE router.

This configuration must be done only in the primary PE router and is used for outbound BGP updates for the next hops.

```
[edit routing-instance]
routing-instance-name {
    egress-protection {
        context-identifier {
            context-id-ip-address;
        }
    }
}
```

Configuring the `context-identifier` statement at the `[edit routing-instances routing-instance-name]` hierarchy level provides customer edge VRF-level context ID granularity for each VRF instance.

- `context-identifier`—This statement specifies an IPV4 address used to define the pair of PE routers participating in the egress protection LSP. The context identifier is used to assign an identifier to the protector PE router. The identifier is propagated to the other PE routers participating in the network, making it possible for the protected egress PE router to signal the egress protection LSP to the protector PE router.

Configuration

IN THIS SECTION

CLI Quick Configuration

NOTE: This example only shows sample configuration that is relevant to configuring egress PE protection for Layer 3 VPN services on the protected router, PE2, the protector router, PE3, and the PLR router.

To quickly configure this example, copy the following commands, paste them into a text file, remove any line breaks, change any details necessary to match your network configuration, and then copy and paste the commands into the CLI at the `[edit]` hierarchy level.

PE2 (Protected PE Router)

```
set protocols mpls interface all
set protocols mpls interface fxp0.0 disable
set protocols mpls egress-protection context-identifier 192.0.2.6 primary
set protocols bgp group ibgp type internal
set protocols bgp group ibgp local-address 10.255.245.194
set protocols bgp group ibgp family inet-vpn unicast egress-protection context-identifier
192.0.2.6
```

PE3 (Protector PE Router)

```
set protocols mpls interface all
set protocols mpls interface fxp0.0 disable
set protocols mpls egress-protection context-identifier 192.0.2.6 protector
set protocols bgp group ibgp type internal
set protocols bgp group ibgp local-address 10.255.245.196
set protocols bgp group ibgp family inet-vpn unicast egress-protection keep-import remote-vrf
set policy-options policy-statement remote-vrf from community rsite1
set policy-options policy-statement remote-vrf from community rsite24
set policy-options policy-statement remote-vrf then accept
set policy-options community rsite1 members target:1:1
set policy-options community rsite24 members target:100:1023
```

PLR Router

```
set protocols mpls interface all
set protocols mpls interface fxp0.0 disable
set protocols isis level 1 disable
set protocols isis interface all node-link-protection
```

```
set protocols isis backup-spf-options per-prefix-calculation
set protocols ldp track-igp-metric
set protocols ldp interface all
set protocols ldp interface fxp0.0 disable
```

Configuring the Protected PE Router (PE2)

Step-by-Step Procedure

To configure the protected PE router, PE2:

1. Configure MPLS on the interfaces.

   ```
   [edit protocols mpls]
   user@PE2# set interface all
   user@PE2#set interface fxp0.0 disable
   ```

2. Configure egress protection and the context identifier.

   NOTE: The context identifier type must be set to `primary`.

   ```
   [edit protocols mpls]
   user@PE2# set egress-protection context-identifier 192.0.2.6 primary
   ```

3. Configure egress protection for the configured BGP NRLI.

   NOTE: The context identifier configured at the `[edit protocols bgp group group-name family inet-vpn]` hierarchy level should match the context identifier configured at the `[edit protocols mpls]` hierarchy level.

   ```
   [edit protocols bgp]
   user@PE2# set group ibgp type internal
   user@PE2# set group ibgp local-address 10.255.245.194
   user@PE2# set group ibgp family inet-vpn unicast egress-protection context-identifier
   192.0.2.6
   ```

NOTE: Configuring the context-identifier at the `[edit routing-instances routing-instance-name]` hierarchy level provides CE VRF-level context-id granularity for each virtual routing and forwarding (VRF) instance.

4. After you are done configuring the device, commit the configuration.

```
[edit]
user@PE2#  commit
```

Results

Confirm your configuration by issuing the show protocols command. If the output does not display the intended configuration, repeat the instructions in this example to correct the configuration.

```
user@PE2# show protocols
mpls {
    interface all;
    interface fxp0.0 {
        disable;
    }
    egress-protection {
        context-identifier 192.0.2.6 {
            primary;
        }
    }
}
    bgp {
        group ibgp {
            type internal;
            local-address 10.255.245.194;
            family inet-vpn {
                unicast {
                    egress-protection {
                        context-identifier {
                            192.0.2.6;
                        }
                    }
                }
            }
```

```
        }
    }
```

Configuring the Protector PE Router (PE3)

Step-by-Step Procedure

To configure the protector PE router, PE3:

1. Configure MPLS on the interfaces.

   ```
   [edit protocols mpls]
   user@PE3# set interface all
   user@PE3#set mpls interface fxp0.0 disable
   ```

2. Configure egress protection and the context identifier.

   ```
   [edit protocols mpls]
   user@PE3#set egress-protection context-identifier 192.0.2.6 protector
   ```

3. Configure IPv4 Layer 3 VPN NRLI parameters.

   ```
   [edit protocols bgp]
   user@PE3# set group ibgp type internal
   user@PE3# set group ibgp local-address 10.255.245.196
   user@PE3# set group ibgp family inet-vpn unicast egress-protection keep-import remote-vrf
   ```

4. Configure routing policy options.

   ```
   [edit policy-options]
   user@PE3# set policy-statement remote-vrf from community rsite1
   user@PE3# set policy-statement remote-vrf from community rsite24
   user@PE3# set policy-statement remote-vrf then accept
   user@PE3# set community rsite1 members target:1:1
   user@PE3# set community rsite24 members target:100:1023
   ```

5. After you are done configuring the device, commit the configuration.

```
[edit]
user@PE3#  commit
```

Results

Confirm your configuration by issuing the `show protocols` and the `show policy-options` commands. If the output does not display the intended configuration, repeat the instructions in this example to correct the configuration.

```
user@PE3# show protocols
mpls {
    interface all;
    interface fxp0.0 {
        disable;
    }
    egress-protection {
        context-identifier 192.0.2.6 {
            protector;
        }
    }
}
    bgp {
        group ibgp {
            type internal;
            local-address 10.255.245.196;
            family inet-vpn {
                unicast {
                    egress-protection {
                        keep-import remote-vrf;
                    }
                }
            }
        }
    }
```

```
user@PE3# show policy-options
policy-statement remote-vrf {
```

```
    from community [ rsite1 rsite24 ];
    then accept;
    }
    community rsite1 members target:1:1;
community rsite24 members target:100:1023;
```

Configuring the PLR Router

Step-by-Step Procedure

To configure the router acting as the point of local repair (PLR):

1. Configure MPLS on the interfaces.

   ```
   [edit protocols mpls]
   user@PLR# set interface all
   user@PLR# set interface fxp0.0 disable
   ```

2. Configure per-prefix-LFA calculation along with link protection.

   ```
   [edit protocols isis]
   user@PLR# set backup-spf-options per-prefix-calculation
   user@PLR# set level 1 disable
   user@PLR# set interface all node-link-protection
   user@PLR# set interface fxp0.0 disable
   ```

3. Configure LDP to use the interior gateway protocol (IGP) route metric instead of the default LDP route metric (the default LDP route metric is 1).

   ```
   [edit protocols ldp]
   user@PLR# set track-igp-metric
   user@PLR# set interface all
   user@PLR# set interface fxp0.0 disable
   ```

Results

Confirm your configuration by issuing the `show protocols` command. If the output does not display the intended configuration, repeat the instructions in this example to correct the configuration.

```
user@PLR# show protocols
mpls {
    interface all;
    interface fxp0.0 {
        disable;
    }
}
isis {
    backup-spf-options per-prefix-calculation;
    level 1 disable;
    interface all {
        node-link-protection;
    }
}
ldp {
    track-igp-metric;
    interface all;
    interface fxp0.0 {
        disable;
    }
}
```

Verification

IN THIS SECTION

Confirm that the configuration is working properly.

Verifying Egress Protection Details

Purpose

Check the egress protection configuration.

Action

```
user@PE3> show mpls egress-protection details

Instance                 Type      Protection-Type
rsite1                  remote-vrf  Protector
  RIB __192.0.2.6-rsite1__.inet.0, Context-Id 192.0.2.6, Enhanced-lookup
  Route Target 1:1
rsite24                 remote-vrf  Protector
  RIB __192.0.2.6-rsite24__.inet.0, Context-Id 192.0.2.6, Enhanced-lookup
  Route Target 100:1023
```

Meaning

`Instance` indicates the routing-instance name. `Type` shows the type of the VRF. It can be either `local-vrf` or `remote-vrf`. `RIB` (routing information base) indicates the edge-protection created routing table. `Context-Id` shows the context ID associated with the RIB. `Route Target` shows the route target associated with the routing instance.

Verifying Routing Instances

Purpose

Verify the routing instances.

Action

```
user@PE3> show route instance site1 detail

site1:
  Router ID: 198.51.100.1
  Type: vrf               State: Active
  Interfaces:
    lt-1/3/0.8
```

```
  Route-distinguisher: 10.255.255.11:150
  Vrf-import: [ site1-import ]
  Vrf-export: [ __vrf-export-site1-internal__ ]
  Vrf-export-target: [ target:100:250 ]
  Fast-reroute-priority: low
  Vrf-edge-protection-id:  192.0.2.6
  Tables:
    site1.inet.0           : 27 routes (26 active, 0 holddown, 0 hidden)
    site1.iso.0            : 0 routes (0 active, 0 holddown, 0 hidden)
    site1.inet6.0          : 0 routes (0 active, 0 holddown, 0 hidden)
    site1.mdt.0            : 0 routes (0 active, 0 holddown, 0 hidden)
```

Meaning

`Vrf-edge-protection-id` shows the egress protection configured in the protector PE router with the routing instance.

Verifying BGP NRLI

Purpose

Check the details of the BGP VPN network layer reachability information.

Action

```
user@PE3> show bgp neighbor

Peer: 10.255.55.1+179 AS 65535 Local: 10.255.22.1+59264 AS 65535
  Type: Internal    State: Established    Flags: <ImportEval Sync>
  Last State: OpenConfirm   Last Event: RecvKeepAlive
  Last Error: None
  Options: <Preference LocalAddress KeepAll AddressFamily Rib-group Refresh>
  Address families configured: inet-vpn-unicast
  Local Address: 10.255.22.1 Holdtime: 90 Preference: 170
  NLRI configured with egress-protection: inet-vpn-unicast
  Egress-protection NLRI inet-vpn-unicast, keep-import: [ VPN-A-remote ]
  Number of flaps: 0
```

Meaning

`NLRI configured with egress-protection` shows the BGP family configured with egress protection. `egress-protection NLRI inet-vpn-unicast, keep-import: [remote-vrf]` shows the egress protection routing policy for the BGP group.

Example: Configuring Layer 3 VPN Egress Protection with RSVP and LDP

IN THIS SECTION

This example shows how to configure fast service restoration at the egress of a Layer 3 VPN when the customer is multihomed to the service provider. Further, this example includes enhanced point-of-local-repair (PLR) functionality, in which the PLR reroutes service traffic during an egress failure.

Starting in Junos OS Release 13.3, enhanced PLR functionality is available, in which the PLR reroutes service traffic during an egress failure. As part of this enhancement, the PLR router no longer needs to be directly connected to the protector router. Previously, if the PLR was not directly connected to the protector router, the loop-free alternate route could not find the backup path to the protector.

Requirements

No special configuration beyond device initialization is required before configuring this example.

This example requires Junos OS Release 13.3 or later.

Overview

IN THIS SECTION

In this example, the customer edge (CE) devices are part of a VPN where Device CE1 is multihomed with Device PE2 and Device PE3.

Device PE3 acts as the protector for the Layer 3 VPN routing instances or subnets.

Device PE1 is the originator for the context identifier for Device CE1, Device PE2 is the primary router for that context identifier, while Device PE3 is the protector for that context identifier.

Device P1 acts as the point of local repair (PLR). As such, Device P1 can redirect Layer 3 VPN traffic to the protector PE router to enable fast restoration and reroute.

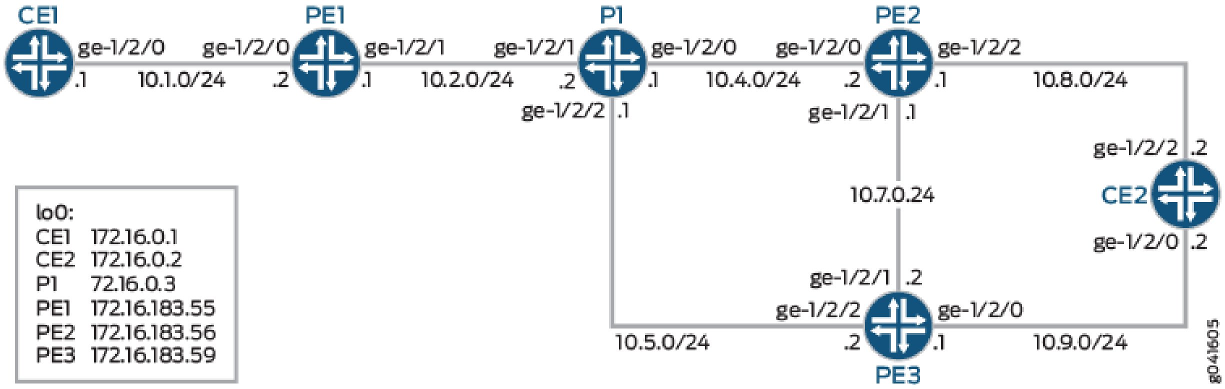
The working path is through P1>PE2. The backup path is through P1>PE3. Traffic flows through the working path under normal circumstances. When a Device PE2 node or link failure is detected, traffic is rerouted from the working path to the protected path. In the normal failover process, the detection of failure and the recovery rely on the control plane and is therefore relatively slow. Typically, if there is a link or node failure in the core network, the egress PE router would have to rely on the ingress PE router to detect the failure and switch over to the backup path, because a local repair option for egress failure is not available. To provide a local repair solution for the egress PE link or node failure, a mechanism known as egress protection is used in this example to repair and restore the connection quickly. Because egress protection is configured, the PLR router detects the Device PE2 link or node failure and reroutes traffic through the protector Device PE3 using the backup LDP-signaled label-switched path (LSP). The PLR router uses per-prefix loop-free alternate routes to program the backup next hop through Device PE3, and traffic is forwarded to Device CE2 using the alternate paths. This restoration is done quickly after the PLR router detects the Device PE2 egress node or link failure. The dual protection mechanism can also be used for egress protection where the two PE routers can simultaneously act as the primary PE router and the protector PE router for their respective context ID routes or next hops.

In addition to egress protection, this example demonstrates an enhanced PLR function, in which the PLR reroutes service traffic during the egress failure. This enhancement is supported in Junos OS Release 13.3 and later. In this example, Device P1 (the PLR) is directly connected to Device PE3 (the protector). A new configuration statement, `advertise-mode`, enables you to set the method for the interior gateway protocol (IGP) to advertise egress protection availability.

Topology

Figure 82 on page 1023 shows the sample network.

Figure 82: Layer 3 VPN Egress Protection with RSVP and LDP



Configuration

IN THIS SECTION

CLI Quick Configuration

To quickly configure this example, copy the following commands, paste them into a text file, remove any line breaks, change any details necessary to match your network configuration, and then copy and paste the commands into the CLI at the `[edit]` hierarchy level.

Device CE1

```
set interfaces ge-1/2/0 unit 0 description to_PE1
set interfaces ge-1/2/0 unit 0 family inet address 10.1.0.1/24
set interfaces lo0 unit 0 family inet address 172.16.0.1/32
set protocols ospf area 0.0.0.0 interface ge-1/2/0.0
```

Device CE2

```
set interfaces ge-1/2/2 unit 0 description to_PE2
set interfaces ge-1/2/2 unit 0 family inet address 10.8.0.2/24
set interfaces ge-1/2/0 unit 0 description to_PE3
```

```
set interfaces ge-1/2/0 unit 0 family inet address 10.9.0.2/24
set interfaces lo0 unit 0 family inet address 172.16.0.2/32
set protocols ospf area 0.0.0.0 interface ge-1/2/2.0
set protocols ospf area 0.0.0.0 interface ge-1/2/0.0
```

Device P1

```
set interfaces ge-1/2/1 unit 0 description to_PE1
set interfaces ge-1/2/1 unit 0 family inet address 10.2.0.2/24
set interfaces ge-1/2/1 unit 0 family iso
set interfaces ge-1/2/1 unit 0 family mpls
set interfaces ge-1/2/0 unit 0 description to_PE2
set interfaces ge-1/2/0 unit 0 family inet address 10.4.0.1/24
set interfaces ge-1/2/0 unit 0 family iso
set interfaces ge-1/2/0 unit 0 family mpls
set interfaces ge-1/2/2 unit 0 description to_PE3
set interfaces ge-1/2/2 unit 0 family inet address 10.5.0.1/24
set interfaces ge-1/2/2 unit 0 family iso
set interfaces ge-1/2/2 unit 0 family mpls
set interfaces lo0 unit 0 family inet address 172.16.0.3/32
set interfaces lo0 unit 0 family iso address 49.0002.0172.0016.0003.00
set protocols rsvp interface all
set protocols rsvp interface fxp0.0 disable
set protocols mpls interface all
set protocols isis backup-spf-options per-prefix-calculation
set protocols isis level 1 disable
set protocols isis level 2 wide-metrics-only
set protocols isis interface all node-link-protection
set protocols isis interface fxp0.0 disable
set protocols isis interface lo0.0
set protocols ldp track-igp-metric
set protocols ldp interface all
set protocols ldp interface fxp0.0 disable
```

Device PE1

```
set interfaces ge-1/2/0 unit 0 description to_CE1
set interfaces ge-1/2/0 unit 0 family inet address 10.1.0.2/24
set interfaces ge-1/2/1 unit 0 description to_P1
set interfaces ge-1/2/1 unit 0 family inet address 10.2.0.1/24
set interfaces ge-1/2/1 unit 0 family iso
```

```
set interfaces ge-1/2/1 unit 0 family mpls
set interfaces lo0 unit 0 family inet address 172.16.183.55/32
set interfaces lo0 unit 0 family iso address 49.0002.1720.1618.3055.00
set protocols rsvp interface all
set protocols rsvp interface fxp0.0 disable
set protocols mpls label-switched-path toPrimary192.0.2.6 to 192.0.2.6
set protocols mpls label-switched-path toPrimary192.0.2.6 egress-protection
set protocols mpls interface all
set protocols bgp group ibgp type internal
set protocols bgp group ibgp local-address 172.16.183.55
set protocols bgp group ibgp family inet-vpn unicast
set protocols bgp group ibgp neighbor 172.16.183.56
set protocols bgp group ibgp neighbor 172.16.183.59
set protocols isis level 1 disable
set protocols isis level 2 wide-metrics-only
set protocols isis interface all
set protocols isis interface fxp0.0 disable
set protocols isis interface lo0.0
set protocols ldp track-igp-metric
set protocols ldp interface all
set protocols ldp interface fxp0.0 disable
set routing-instances vpn1 instance-type vrf
set routing-instances vpn1 interface ge-1/2/0.0
set routing-instances vpn1 route-distinguisher 172.16.183.55:10
set routing-instances vpn1 vrf-target target:10:10
set routing-instances vpn1 routing-options static route 100.0.0.0/24 next-hop 10.1.0.1
set routing-instances vpn1 protocols ospf area 0.0.0.0 interface ge-1/2/0.0
set routing-options autonomous-system 64510
```

Device PE2

```
set interfaces ge-1/2/0 unit 0 description to_P1
set interfaces ge-1/2/0 unit 0 family inet address 10.4.0.2/24
set interfaces ge-1/2/0 unit 0 family iso
set interfaces ge-1/2/0 unit 0 family mpls
set interfaces ge-1/2/2 unit 0 description to_CE2
set interfaces ge-1/2/2 unit 0 family inet address 10.8.0.1/24
set interfaces ge-1/2/1 unit 0 description to_PE3
set interfaces ge-1/2/1 unit 0 family inet address 10.7.0.1/24
set interfaces ge-1/2/1 unit 0 family iso
set interfaces ge-1/2/1 unit 0 family mpls
set interfaces lo0 unit 0 family inet address 172.16.183.56/32
```

```
set interfaces lo0 unit 0 family iso address 49.0002.1720.1618.3056.00
set protocols rsvp interface all
set protocols rsvp interface fxp0.0 disable
set protocols mpls label-switched-path toPE1 to 172.16.183.55
set protocols mpls label-switched-path toPrimary192.0.2.6 to 192.0.2.6
set protocols mpls label-switched-path toPrimary192.0.2.6 egress-protection
set protocols mpls interface all
set protocols mpls egress-protection context-identifier 192.0.2.6 primary
set protocols mpls egress-protection context-identifier 192.0.2.6 advertise-mode stub-proxy
set protocols bgp group ibgp type internal
set protocols bgp group ibgp local-address 172.16.183.56
set protocols bgp group ibgp family inet-vpn unicast egress-protection context-identifier
192.0.2.6
set protocols bgp group ibgp neighbor 172.16.183.55
set protocols bgp group ibgp neighbor 172.16.183.59
set protocols isis level 1 disable
set protocols isis level 2 wide-metrics-only
set protocols isis interface all
set protocols isis interface fxp0.0 disable
set protocols isis interface lo0.0
set protocols ldp track-igp-metric
set protocols ldp interface all
set protocols ldp interface fxp0.0 disable
set routing-options autonomous-system 64510
```

Device PE3

```
set interfaces ge-1/2/2 unit 0 description to_P1
set interfaces ge-1/2/2 unit 0 family inet address 10.5.0.2/24
set interfaces ge-1/2/2 unit 0 family iso
set interfaces ge-1/2/2 unit 0 family mpls
set interfaces ge-1/2/0 unit 0 description to_CE2
set interfaces ge-1/2/0 unit 0 family inet address 10.9.0.1/24
set interfaces ge-1/2/1 unit 0 description to_PE2
set interfaces ge-1/2/1 unit 0 family inet address 10.7.0.2/24
set interfaces ge-1/2/1 unit 0 family iso
set interfaces ge-1/2/1 unit 0 family mpls
set interfaces lo0 unit 0 family inet address 172.16.183.59/32
set interfaces lo0 unit 0 family iso address 49.0002.1720.1618.3059.00
set protocols rsvp interface all
set protocols rsvp interface fxp0.0 disable
set protocols mpls label-switched-path toPE1 to 172.16.183.55
```

```
set protocols mpls interface all
set protocols mpls egress-protection context-identifier 192.0.2.6 protector
set protocols mpls egress-protection context-identifier 192.0.2.6 advertise-mode stub-proxy
set protocols bgp group ibgp type internal
set protocols bgp group ibgp local-address 172.16.183.59
set protocols bgp group ibgp family inet-vpn unicast egress-protection keep-import remote-vrf
set protocols bgp group ibgp neighbor 172.16.183.55
set protocols bgp group ibgp neighbor 172.16.183.56
set protocols isis level 1 disable
set protocols isis level 2 wide-metrics-only
set protocols isis interface all
set protocols isis interface fxp0.0 disable
set protocols isis interface lo0.0
set protocols ldp track-igp-metric
set protocols ldp interface all
set policy-options policy-statement remote-vrf from community rsite1
set policy-options policy-statement remote-vrf from community rsite24
set policy-options policy-statement remote-vrf then accept
set policy-options community rsite1 members target:1:1
set policy-options community rsite24 members target:100:1023
set routing-options autonomous-system 64510
```

Procedure

Step-by-Step Procedure

The following example requires that you navigate various levels in the configuration hierarchy. For information about navigating the CLI, see *Using the CLI Editor in Configuration Mode* in the CLI User Guide.

To configure Device P1 (the PLR):

1. Configure the device interfaces.

```
[edit interfaces]
user@P1# set ge-1/2/1 unit 0 description to_PE1
user@P1# set ge-1/2/1 unit 0 family inet address 10.2.0.2/24
user@P1# set ge-1/2/1 unit 0 family iso
user@P1# set ge-1/2/1 unit 0 family mpls
user@P1# set ge-1/2/0 unit 0 description to_PE2
user@P1# set ge-1/2/0 unit 0 family inet address 10.4.0.1/24
user@P1# set ge-1/2/0 unit 0 family iso
```

```
user@P1# set ge-1/2/0 unit 0 family mpls
user@P1# set ge-1/2/2 unit 0 description to_PE3
user@P1# set ge-1/2/2 unit 0 family inet address 10.5.0.1/24
user@P1# set ge-1/2/2 unit 0 family iso
user@P1# set ge-1/2/2 unit 0 family mpls
user@P1# set lo0 unit 0 family inet address 172.16.0.3/32
user@P1# set lo0 unit 0 family iso address 49.0002.0172.0016.0003.00
```

2. Configure IS-IS.

   Configure per-prefix-LFA calculation along with node link protection.

```
[edit protocols isis]
user@P1# set backup-spf-options per-prefix-calculation
user@P1# set level 1 disable
user@P1# set level 2 wide-metrics-only
user@P1# set interface all node-link-protection
user@P1# set interface fxp0.0 disable
user@P1# set interface lo0.0
```

3. Enable MPLS.

```
[edit protocols mpls ]
user@P1# set interface all
```

4. Enable RSVP.

```
[edit protocols rsvp]
user@P1# set interface all
user@P1# set interface fxp0.0 disable
```

5. Enable LDP.

```
[edit protocols ldp]
user@P1# set track-igp-metric
user@P1# set interface all
user@P1# set interface fxp0.0 disable
```

Step-by-Step Procedure

The following example requires that you navigate various levels in the configuration hierarchy. For information about navigating the CLI, see *Using the CLI Editor in Configuration Mode* in the CLI User Guide.

To configure Device PE1:

1. Configure the device interfaces.

   ```
   [edit interfaces]
   user@PE1# set ge-1/2/0 unit 0 description to_CE1
   user@PE1# set ge-1/2/0 unit 0 family inet address 10.1.0.2/24
   user@PE1# set ge-1/2/1 unit 0 description to_P1
   user@PE1# set ge-1/2/1 unit 0 family inet address 10.2.0.1/24
   user@PE1# set ge-1/2/1 unit 0 family iso
   user@PE1# set ge-1/2/1 unit 0 family mpls
   user@PE1# set lo0 unit 0 family inet address 172.16.183.55/32
   user@PE1# set lo0 unit 0 family iso address 49.0002.1720.1618.3055.00
   ```

2. Enable RSVP.

   ```
   [edit protocols rsvp]
   user@PE1# set interface all
   user@PE1# set interface fxp0.0 disable
   ```

3. Configure MPLS.

   ```
   [edit protocols mpls]
   user@PE1# set label-switched-path toPrimary192.0.2.6 to 192.0.2.6
   user@PE1# set label-switched-path toPrimary192.0.2.6 egress-protection
   user@PE1# set interface all
   ```

4. Configure IBGP.

   ```
   [edit protocols bgp group ibgp]
   user@PE1# set type internal
   user@PE1# set local-address 172.16.183.55
   user@PE1# set family inet-vpn unicast
   ```

```
user@PE1# set neighbor 172.16.183.56
user@PE1# set neighbor 172.16.183.59
```

5. Configure IS-IS.

```
[edit protocols isis]
user@PE1# set level 1 disable
user@PE1# set level 2 wide-metrics-only
user@PE1# set interface all
user@PE1# set interface fxp0.0 disable
user@PE1# set interface lo0.0
```

6. Enable LDP.

```
[edit protocols ldp]
user@PE1# set track-igp-metric
user@PE1# set interface all
user@PE1# set interface fxp0.0 disable
```

7. Configure the routing instance.

```
[edit routing-instances vpn1]
user@PE1# set instance-type vrf
user@PE1# set interface ge-1/2/0.0
user@PE1# set route-distinguisher 172.16.183.55:10
user@PE1# set vrf-target target:10:10
user@PE1# set routing-options static route 100.0.0.0/24 next-hop 10.1.0.1
user@PE1# set protocols ospf area 0.0.0.0 interface ge-1/2/0.0
```

8. Configure the autonomous system (AS) number.

```
[edit routing-options]
user@PE1# set autonomous-system 64510
```

Step-by-Step Procedure

The following example requires that you navigate various levels in the configuration hierarchy. For information about navigating the CLI, see *Using the CLI Editor in Configuration Mode* in theCLI User Guide.

To configure Device PE2:

1. Configure the device interfaces.

```
[edit interfaces]
user@PE2# set ge-1/2/0 unit 0 description to_P1
user@PE2# set ge-1/2/0 unit 0 family inet address 10.4.0.2/24
user@PE2# set ge-1/2/0 unit 0 family iso
user@PE2# set ge-1/2/0 unit 0 family mpls
user@PE2# set ge-1/2/2 unit 0 description to_CE2
user@PE2# set ge-1/2/2 unit 0 family inet address 10.8.0.1/24
user@PE2# set ge-1/2/1 unit 0 description to_PE3
user@PE2# set ge-1/2/1 unit 0 family inet address 10.7.0.1/24
user@PE2# set ge-1/2/1 unit 0 family iso
user@PE2# set ge-1/2/1 unit 0 family mpls
user@PE2# set lo0 unit 0 family inet address 172.16.183.56/32
user@PE2# set lo0 unit 0 family iso address 49.0002.1720.1618.3056.00
```

2. Enable RSVP.

```
[edit protocols rsvp]
user@PE2# set interface all
user@PE2# set interface fxp0.0 disable
```

3. Configure MPLS.

```
[edit protocols mpls]
user@PE2# set label-switched-path toPE1 to 172.16.183.55
user@PE2# set label-switched-path toPrimary192.0.2.6 to 192.0.2.6
user@PE2# set label-switched-path toPrimary192.0.2.6 egress-protection
user@PE2# set interface all
user@PE2# set egress-protection context-identifier 192.0.2.6 primary
user@PE2# set egress-protection context-identifier 192.0.2.6 advertise-mode stub-proxy
```

4. Configure IBGP.

```
[edit protocols bgp group ibgp]
user@PE2# set type internal
user@PE2# set local-address 172.16.183.56
user@PE2# set family inet-vpn unicast egress-protection context-identifier 192.0.2.6
user@PE2# set neighbor 172.16.183.55
user@PE2# set neighbor 172.16.183.59
```

5. Configure IS-IS.

```
[edit protocols isis]
user@PE2# set level 1 disable
user@PE2# set level 2 wide-metrics-only
user@PE2# set interface all
user@PE2# set interface fxp0.0 disable
user@PE2# set interface lo0.0
```

6. Enable LDP.

```
[edit protocols ldp]
user@PE2# set track-igp-metric
user@PE2# set interface all
user@PE2# set interface fxp0.0 disable
```

7. Configure the AS number.

```
[edit routing-options]
user@PE2# set autonomous-system 64510
```

Step-by-Step Procedure

The following example requires that you navigate various levels in the configuration hierarchy. For information about navigating the CLI, see *Using the CLI Editor in Configuration Mode* in the CLI User Guide.

To configure Device PE3:

1. Configure the device interfaces.

```
[edit interfaces]
user@PE3# set ge-1/2/2 unit 0 description to_P1
user@PE3# set ge-1/2/2 unit 0 family inet address 10.5.0.2/24
user@PE3# set ge-1/2/2 unit 0 family iso
user@PE3# set ge-1/2/2 unit 0 family mpls
user@PE3# set ge-1/2/0 unit 0 description to_CE2
user@PE3# set ge-1/2/0 unit 0 family inet address 10.9.0.1/24
user@PE3# set ge-1/2/1 unit 0 description to_PE2
user@PE3# set ge-1/2/1 unit 0 family inet address 10.7.0.2/24
user@PE3# set ge-1/2/1 unit 0 family iso
user@PE3# set ge-1/2/1unit 0 family mpls
user@PE3# set lo0 unit 0 family inet address 172.16.183.59/32
user@PE3# set lo0 unit 0 family iso address 49.0002.1720.1618.3059.00
```

2. Enable RSVP.

```
[edit protocols rsvp]
user@PE3# set interface all
user@PE3# set interface fxp0.0 disable
```

3. Configure MPLS.

```
[edit protocols mpls]
user@PE3# set label-switched-path toPE1 to 172.16.183.55
user@PE3# set interface all
user@PE3# set egress-protection context-identifier 192.0.2.6 protector
user@PE3# set egress-protection context-identifier 192.0.2.6 advertise-mode stub-proxy
```

4. Configure IBGP.

```
[edit protocols bgp group ibgp]
user@PE3# set type internal
user@PE3# set local-address 172.16.183.59
user@PE3# set family inet-vpn unicast egress-protection keep-import remote-vrf
user@PE3# set neighbor 172.16.183.55
user@PE3# set neighbor 172.16.183.56
```

5. Configure IS-IS.

```
[edit protocols isis]
user@PE3# set level 1 disable
user@PE3# set level 2 wide-metrics-only
user@PE3# set interface all
user@PE3# set interface fxp0.0 disable
user@PE3# set interface lo0.0
```

6. Enable LDP.

```
[edit protocols ldp]
user@PE3# set track-igp-metric
user@PE3# set interface all
```

7. Configure the routing policy.

```
[edit policy-options]
user@PE3# set policy-statement remote-vrf from community rsite1
user@PE3# set policy-statement remote-vrf from community rsite24
user@PE3# set policy-statement remote-vrf then accept
user@PE3# set community rsite1 members target:1:1
user@PE3# set community rsite24 members target:100:1023
```

8. Configure the AS number.

```
[edit routing-options]
user@PE3# set autonomous-system 64510
```

Results

From configuration mode, confirm your configuration by entering the `show interfaces` and `show protocols` commands. If the output does not display the intended configuration, repeat the instructions in this example to correct the configuration.

Device P1

```
user@P1# show interfaces
ge-1/2/0 {
    unit 0 {
        description to_PE2;
        family inet {
            address 10.4.0.1/24;
        }
        family iso;
        family mpls;
    }
}
ge-1/2/1 {
    unit 0{
        description to_PE1;
        family inet {
            address 10.2.0.2/24;
        }
        family iso;
        family mpls;
    }
}
ge-1/2/2 {
    unit 0 {
        description to_PE3;
        family inet {
            address 10.5.0.1/24;
        }
        family iso;
        family mpls;
    }
}
lo0 {
    unit 0 {
        family inet {
            address 172.16.0.3/32;
        }
        family iso {
            address 49.0002.0172.0016.0003.00;
        }
```

```
    }
}
```

```
user@P1# show protocols
rsvp {
    interface all;
    interface fxp0.0 {
        disable;
    }
}
mpls {
    interface all;
}
isis {
    backup-spf-options per-prefix-calculation;
    level 1 disable;
    level 2 wide-metrics-only;
    interface all {
        node-link-protection;
    }
    interface fxp0.0 {
        disable;
    }
    interface lo0.0;
}
ldp {
    track-igp-metric;
    interface all;
    interface fxp0.0 {
        disable;
    }
}
```

Device PE1

```
user@PE1# show interfaces
ge-1/2/0 {
    unit 0 {
        description to_CE1;
        family inet {
            address 10.1.0.2/24;
```

```
        }
    }
}
ge-1/2/1 {
    unit 0 {
        description to_P1;
        family inet {
            address 10.2.0.1/24;
        }
        family iso;
        family mpls;
    }
}
lo0 {
    unit 0 {
        family inet {
            address 172.16.183.55/32;
        }
        family iso {
            address 49.0002.1720.1618.3055.00;
        }
    }
}
```

```
user@PE1# show protocols
rsvp {
    interface all;
    interface fxp0.0 {
        disable;
    }
}
mpls {
    label-switched-path toPE2Primary192.0.2.6 {
        to 192.0.2.6;
        egress-protection;
    }
    interface all;
}
bgp {
    group ibgp {
        type internal;
```

```
        local-address 172.16.183.55;
        family inet-vpn {
            unicast;
        }
        neighbor 172.16.183.56;
        neighbor 172.16.183.59;
    }
}
isis {
    level 1 disable;
    level 2 wide-metrics-only;
    interface all;
    interface fxp0.0 {
        disable;
    }
    interface lo0.0;
}
ldp {
    track-igp-metric;
    interface all;
    interface fxp0.0 {
        disable;
    }
}
```

```
user@PE1# show routing-instances
vpn1 {
    instance-type vrf;
    interface ge-1/2/0.0;
    route-distinguisher 172.16.183.55:10;
    vrf-target target:10:10;
    routing-options {
        static {
            route 100.0.0.0/24 next-hop 10.1.0.1;
        }
    }
    protocols {
        ospf {
            area 0.0.0.0 {
                interface ge-1/2/0.0;
            }
```

```
        }
    }
}
```

```
user@PE1# show routing-options
autonomous-system 64510;
```

Device PE2

```
user@PE2# show interfaces
ge-1/2/0 {
    unit 0 {
        description to_P1;
        family inet {
            address 10.4.0.2/24;
        }
        family iso;
        family mpls;
    }
}
ge-1/2/1 {
    unit 0 {
        description to_PE3;
        family inet {
            address 10.7.0.1/24;
        }
        family iso;
        family mpls;
    }
}
ge-1/2/2 {
    unit 0 {
        description to_CE2;
        family inet {
            address 10.8.0.1/24;
        }
    }
}
lo0 {
    unit 0 {
        family inet {
```

```
            address 172.16.183.56/32;
        }
        family iso {
            address 49.0002.1720.1618.3056.00;
        }
    }
}
```

```
user@PE2# show protocols
rsvp {
    interface all;
    interface fxp0.0 {
        disable;
    }
}
mpls {
    label-switched-path toPE1 {
        to 172.16.183.55;
    }
    label-switched-path toPE2Primary192.0.2.6 {
        to 192.0.2.6;
        egress-protection;
    }
    interface all;
    egress-protection {
        context-identifier 192.0.2.6 {
            primary;
            advertise-mode stub-proxy;
        }
    }
}
bgp {
    group ibgp {
        type internal;
        local-address 172.16.183.56;
        family inet-vpn {
            unicast {
                egress-protection {
                    context-identifier {
                        192.0.2.6;
                    }
```

```
                }
            }
        }
        neighbor 172.16.183.55;
        neighbor 172.16.183.59;
    }
}
isis {
    level 1 disable;
    level 2 wide-metrics-only;
    interface all;
    interface fxp0.0 {
        disable;
    }
    interface lo0.0;
}
ldp {
    track-igp-metric;
    interface all;
    interface fxp0.0 {
        disable;
    }
}
```

```
user@PE2# show routing-options
autonomous-system 64510;
```

Device PE3

```
user@PE3# show interfaces
ge-1/2/0 {
    unit 0 {
        description to_CE2;
        family inet {
            address 10.9.0.1/24;
        }
    }
}
ge-1/2/1 {
    unit 0 {
        description to_PE2;
```

```
        family inet {
            address 10.7.0.2/24;
        }
        family iso;
        family mpls;
    }
}
ge-1/2/2 {
    unit 0 {
        description to_P1;
        family inet {
            address 10.5.0.2/24;
        }
        family iso;
        family mpls;
    }
}
lo0 {
    unit 0 {
        family inet {
            address 172.16.183.59/32;
        }
        family iso {
            address 49.0002.1720.1618.3059.00;
        }
    }
}
```

```
user@PE3# show protocols
rsvp {
    interface all;
    interface fxp0.0 {
        disable;
    }
}
mpls {
    label-switched-path toPE1 {
        to 172.16.183.55;
    }
    interface all;
    egress-protection {
```

```
        context-identifier 192.0.2.6 {
            protector;
            advertise-mode stub-proxy;
        }
    }
}
bgp {
    group ibgp {
        type internal;
        local-address 172.16.183.59;
        family inet-vpn {
            unicast {
                egress-protection {
                    keep-import remote-vrf;
                }
            }
        }
        neighbor 172.16.183.55;
        neighbor 172.16.183.56;
    }
}
isis {
    level 1 disable;
    level 2 wide-metrics-only;
    interface all;
    interface fxp0.0 {
        disable;
    }
    interface lo0.0;
}
ldp {
    track-igp-metric;
    interface all;
}
```

```
user@PE3# show policy-options
policy-statement remote-vrf {
    from community [ rsite1 rsite24 ];
    then accept;
}
```

```
community rsite1 members target:1:1;
community rsite24 members target:100:1023;
```

```
user@PE3# show routing-options
autonomous-system 64510;
```

If you are done configuring the devices, enter `commit` from configuration mode.

Verification

IN THIS SECTION

Confirm that the configuration is working properly.

Verifying the Protector Node

Purpose

On the protector node (Device PE3), check the information about configured egress protection context identifiers.

Action

```
user@PE3> show mpls context-identifer detail protector

ID: 192.0.2.6
  Type: protector, Metric: 16777215, Mode: proxy
  Context table: __PE3:192.0.2.6__.mpls.0
  Context LSPs:
    toPE2Primary192.0.2.6, from: 172.16.183.55
    toPE2Primary192.0.2.6, from: 172.16.183.56

Total 1, Primary 0, Protector 1
```

Meaning

Device PE3 is the protector node for two LSPs configured from Device PE1 (172.16.183.55) and Device PE2 (172.16.183.56).

Verifying the Primary Node

Purpose

On the primary node (Device PE2), check the information about configured egress protection context identifiers.

Action

```
user@PE2> show mpls context-identifer detail primary

ID: 192.0.2.6
  Type: primary, Metric: 1, Mode: proxy

Total 1, Primary 1, Protector 0
```

Meaning

Device PE2 is the primary node.

Checking the Context Identifier Route

Purpose

Examine the information about the contenxt identifier (192.0.2.6).

Action

```
user@PE1> show route 192.0.2.6



inet.0: 10 destinations, 10 routes (10 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

192.0.2.6/24         *[IS-IS/18] 00:53:39, metric 21
                    > to 10.2.0.2 via ge-1/2/1.0

inet.3: 4 destinations, 4 routes (4 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

192.0.2.6/24         *[LDP/9] 00:53:39, metric 21
                    > to 10.2.0.2 via ge-1/2/1.0, Push 299808
user@PE2> show route 192.0.2.6



inet.0: 13 destinations, 14 routes (13 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

192.0.2.6/24         *[MPLS/1] 3d 02:53:37, metric 1
                      Receive
                    [IS-IS/18] 00:06:08, metric 16777224
                    > to 10.7.0.2 via ge-1/2/1.0
user@PE3> show route 192.0.2.6
inet.0: 13 destinations, 14 routes (13 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

192.0.2.6/24         *[MPLS/2] 3d 02:53:36, metric 16777215
                      Receive
                    [IS-IS/18] 3d 02:53:28, metric 11
                    > to 10.7.0.1 via ge-1/2/1.0

user@P1> show route 192.0.2.6
```

```
inet.0: 12 destinations, 12 routes (12 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

192.0.2.6/24         *[IS-IS/18] 00:53:40, metric 11
                    > to 10.4.0.2 via ge-1/2/0.0

inet.3: 4 destinations, 4 routes (4 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

192.0.2.6/24         *[LDP/9] 00:53:40, metric 11
                    > to 10.4.0.2 via ge-1/2/0.0
```

Verifying Egress Protection

Purpose

On Device PE3, check the routes in the routing table.

Action

```
user@PE3> show mpls egress-protection detail
Instance                 Type      Protection-Type
rsite1                  remote-vrf  Protector
  Route Target 1:1
rsite24                 remote-vrf  Protector
  Route Target 100:1023
```

Meaning

`Instance` indicates the community name. `Type` shows the type of the VRF. It can be either `local-vrf` or `remote-vrf`. `Route Target` shows the route target associated with the routing instance.

Verifying the Routing Instance on Device PE1

Purpose

On Device PE1, check the routes in the routing table.

Action

```
user@PE1> show route instance vpn1 detail

vpn1:
  Router ID: 10.1.0.2
  Type: vrf               State: Active
  Interfaces:
    ge-1/2/0.0
  Route-distinguisher: 172.16.183.55:10
  Vrf-import: [ __vrf-import-vpn1-internal__ ]
  Vrf-export: [ __vrf-export-vpn1-internal__ ]
  Vrf-import-target: [ target:10:10 ]
  Vrf-export-target: [ target:10:10 ]
  Fast-reroute-priority: low
  Tables:
    vpn1.inet.0            : 4 routes (4 active, 0 holddown, 0 hidden)
```

Verifying the LSPs

Purpose

On all devices, check the LSP information.

Action

```
user@PE1> show mpls lsp extensive

Ingress LSP: 1 sessions

192.0.2.6
  From: 172.16.183.55, State: Up, ActiveRoute: 0, LSPname: toPE2Primary192.0.2.6
  ActivePath:  (primary)
  LSPtype: Static Configured, Penultimate hop popping
  LoadBalance: Random
  Encoding type: Packet, Switching type: Packet, GPID: IPv4
 *Primary                    State: Up
    Priorities: 7 0
    SmartOptimizeTimer: 180
    Computed ERO (S [L] denotes strict [loose] hops): (CSPF metric: 16777234)
```

```
 10.2.0.2 S 10.5.0.2 S 192.0.2.6 S (link-id=2)
    Received RRO (ProtectionFlag 1=Available 2=InUse 4=B/W 8=Node 10=SoftPreempt 20=Node-ID):
          10.2.0.2 10.5.0.2
   17 Jun 10 13:13:04.973 CSPF: computation result accepted  10.2.0.2 10.5.0.2 192.0.2.6(link-
id=2)
   16 Jun 10 13:12:36.155 CSPF failed: no route toward 192.0.2.6[4 times]
   15 Jun 10 13:11:26.269 CSPF: link down/deleted: 0.0.0.0(172.16.183.59:2147618818)
(PE3.00/172.16.183.59)->0.0.0.0(192.0.2.6:2)(PE2-192.0.2.6.00/192.0.2.6)
   14 Jun 10 13:10:11.771 Selected as active path
   13 Jun 10 13:10:11.770 Record Route:  10.2.0.2 10.5.0.2
   12 Jun 10 13:10:11.770 Up
   11 Jun 10 13:10:11.634 Originate Call
   10 Jun 10 13:10:11.634 CSPF: computation result accepted  10.2.0.2 10.5.0.2 192.0.2.6(link-
id=2)
    9 Jun 10 13:10:11.623 Clear Call
    8 Jun 10 13:10:11.622 Deselected as active
    7 Jun  7 11:23:08.224 Selected as active path
    6 Jun  7 11:23:08.224 Record Route:  10.2.0.2 10.5.0.2
    5 Jun  7 11:23:08.223 Up
    4 Jun  7 11:23:08.116 Originate Call
    3 Jun  7 11:23:08.116 CSPF: computation result accepted  10.2.0.2 10.5.0.2 192.0.2.6(link-
id=2)
    2 Jun  7 11:22:38.132 CSPF failed: no route toward 192.0.2.6
    1 Jun  7 11:22:08.607 CSPF: could not determine self[8 times]
  Created: Fri Jun  7 11:18:46 2013
Total 1 displayed, Up 1, Down 0

Egress LSP: 2 sessions

172.16.183.55
  From: 172.16.183.59, LSPstate: Up, ActiveRoute: 0
  LSPname: toPE1, LSPpath: Primary
  Suggested label received: -, Suggested label sent: -
  Recovery label received: -, Recovery label sent: -
  Resv style: 1 FF, Label in: 3, Label out: -
  Time left:  126, Since: Mon Jun 10 13:10:11 2013
  Tspec: rate 0bps size 0bps peak Infbps m 20 M 1500
  Port number: sender 2 receiver 10941 protocol 0
  PATH rcvfrom: 10.2.0.2 (ge-1/2/1.0) 105 pkts
  Adspec: received MTU 1500
  PATH sentto: localclient
  RESV rcvfrom: localclient
  Record route: 10.5.0.2 10.2.0.2 <self>
```

```
172.16.183.55
  From: 172.16.183.56, LSPstate: Up, ActiveRoute: 0
  LSPname: toPE1, LSPpath: Primary
  Suggested label received: -, Suggested label sent: -
  Recovery label received: -, Recovery label sent: -
  Resv style: 1 FF, Label in: 3, Label out: -
  Time left:  156, Since: Mon Jun 10 13:10:11 2013
  Tspec: rate 0bps size 0bps peak Infbps m 20 M 1500
  Port number: sender 2 receiver 59956 protocol 0
  PATH rcvfrom: 10.2.0.2 (ge-1/2/1.0) 105 pkts
  Adspec: received MTU 1500
  PATH sentto: localclient
  RESV rcvfrom: localclient
  Record route: 10.4.0.2 10.2.0.2 <self>
Total 2 displayed, Up 2, Down 0

Transit LSP: 0 sessions
Total 0 displayed, Up 0, Down 0
-----

user@PE2> show mpls lsp extensive
Ingress LSP: 2 sessions

192.0.2.6
  From: 172.16.183.56, State: Up, ActiveRoute: 0, LSPname: toPE2Primary192.0.2.6
  ActivePath:  (primary)
  LSPtype: Static Configured, Penultimate hop popping
  LoadBalance: Random
  Encoding type: Packet, Switching type: Packet, GPID: IPv4
 *Primary                    State: Up
    Priorities: 7 0
    SmartOptimizeTimer: 180
    Computed ERO (S [L] denotes strict [loose] hops): (CSPF metric: 16777224)
 10.7.0.2 S 192.0.2.6 S (link-id=2)
    Received RRO (ProtectionFlag 1=Available 2=InUse 4=B/W 8=Node 10=SoftPreempt 20=Node-ID):
          10.7.0.2
   16 Jun 10 13:13:07.220 CSPF: computation result accepted  10.7.0.2 192.0.2.6(link-id=2)
   15 Jun 10 13:12:38.250 CSPF failed: no route toward 192.0.2.6[4 times]
   14 Jun 10 13:11:26.258 CSPF: link down/deleted: 0.0.0.0(172.16.183.59:2147618818)
(PE3.00/172.16.183.59)->0.0.0.0(192.0.2.6:2)(PE2-192.0.2.6.00/192.0.2.6)
   13 Jun 10 13:10:11.746 Selected as active path
   12 Jun 10 13:10:11.743 Record Route:  10.7.0.2
```

```
    11 Jun 10 13:10:11.742 Up
    10 Jun 10 13:10:11.680 Originate Call
     9 Jun 10 13:10:11.680 CSPF: computation result accepted  10.7.0.2 192.0.2.6(link-id=2)
     8 Jun 10 13:10:11.674 Clear Call
     7 Jun 10 13:10:11.669 Deselected as active
     6 Jun  7 11:23:09.370 Selected as active path
     5 Jun  7 11:23:09.370 Record Route:  10.7.0.2
     4 Jun  7 11:23:09.369 Up
     3 Jun  7 11:23:09.349 Originate Call
     2 Jun  7 11:23:09.349 CSPF: computation result accepted  10.7.0.2 192.0.2.6(link-id=2)
     1 Jun  7 11:22:40.140 CSPF failed: no route toward 192.0.2.6[9 times]
  Created: Fri Jun  7 11:18:46 2013

172.16.183.55
  From: 172.16.183.56, State: Up, ActiveRoute: 0, LSPname: toPE1
  ActivePath:  (primary)
  LSPtype: Static Configured, Penultimate hop popping
  LoadBalance: Random
  Encoding type: Packet, Switching type: Packet, GPID: IPv4
 *Primary                    State: Up
    Priorities: 7 0
    SmartOptimizeTimer: 180
    Computed ERO (S [L] denotes strict [loose] hops): (CSPF metric: 20)
 10.4.0.1 S 10.2.0.1 S
    Received RRO (ProtectionFlag 1=Available 2=InUse 4=B/W 8=Node 10=SoftPreempt 20=Node-ID):
          10.4.0.1 10.2.0.1
    13 Jun 10 13:10:11.794 Selected as active path
    12 Jun 10 13:10:11.793 Record Route:  10.4.0.1 10.2.0.1
    11 Jun 10 13:10:11.793 Up
    10 Jun 10 13:10:11.679 Originate Call
     9 Jun 10 13:10:11.679 CSPF: computation result accepted  10.4.0.1 10.2.0.1
     8 Jun 10 13:10:11.660 Clear Call
     7 Jun 10 13:10:11.645 Deselected as active
     6 Jun  7 11:22:40.031 Selected as active path
     5 Jun  7 11:22:40.024 Record Route:  10.4.0.1 10.2.0.1
     4 Jun  7 11:22:40.012 Up
     3 Jun  7 11:22:39.687 Originate Call
     2 Jun  7 11:22:39.687 CSPF: computation result accepted  10.4.0.1 10.2.0.1
     1 Jun  7 11:22:10.235 CSPF failed: no route toward 172.16.183.55[8 times]
  Created: Fri Jun  7 11:18:45 2013
Total 2 displayed, Up 2, Down 0

Egress LSP: 0 sessions
```

```
Total 0 displayed, Up 0, Down 0

Transit LSP: 0 sessions
Total 0 displayed, Up 0, Down 0
user@PE3> show mpls lsp extensive
Ingress LSP: 1 sessions

172.16.183.55
  From: 172.16.183.59, State: Up, ActiveRoute: 0, LSPname: toPE1
  ActivePath:  (primary)
  LSPtype: Static Configured, Penultimate hop popping
  LoadBalance: Random
  Encoding type: Packet, Switching type: Packet, GPID: IPv4
 *Primary                    State: Up
    Priorities: 7 0
    SmartOptimizeTimer: 180
    Computed ERO (S [L] denotes strict [loose] hops): (CSPF metric: 20)
 10.5.0.1 S 10.2.0.1 S
    Received RRO (ProtectionFlag 1=Available 2=InUse 4=B/W 8=Node 10=SoftPreempt 20=Node-ID):
          10.5.0.1 10.2.0.1
   13 Jun 10 13:10:11.708 Selected as active path
   12 Jun 10 13:10:11.703 Record Route:  10.5.0.1 10.2.0.1
   11 Jun 10 13:10:11.703 Up
   10 Jun 10 13:10:11.599 Originate Call
    9 Jun 10 13:10:11.599 CSPF: computation result accepted  10.5.0.1 10.2.0.1
    8 Jun 10 13:10:11.558 Clear Call
    7 Jun 10 13:10:11.555 Deselected as active
    6 Jun  7 11:22:41.829 Selected as active path
    5 Jun  7 11:22:41.828 Record Route:  10.5.0.1 10.2.0.1
    4 Jun  7 11:22:41.827 Up
    3 Jun  7 11:22:41.767 Originate Call
    2 Jun  7 11:22:41.767 CSPF: computation result accepted  10.5.0.1 10.2.0.1
    1 Jun  7 11:22:12.289 CSPF failed: no route toward 172.16.183.55[8 times]
  Created: Fri Jun  7 11:18:45 2013
Total 1 displayed, Up 1, Down 0

Egress LSP: 2 sessions

192.0.2.6
  From: 172.16.183.55, LSPstate: Up, ActiveRoute: 0
  LSPname: toPE2Primary192.0.2.6, LSPpath: Primary
  Suggested label received: -, Suggested label sent: -
  Recovery label received: -, Recovery label sent: -
```

```
  Resv style: 1 FF, Label in: 299920, Label out: 3
  Time left:  141, Since: Mon Jun 10 13:10:11 2013
  Tspec: rate 0bps size 0bps peak Infbps m 20 M 1500
  Port number: sender 2 receiver 17060 protocol 0
  Attrib flags: Non-PHP OOB
  PATH rcvfrom: 10.5.0.1 (ge-1/2/2.0) 105 pkts
  Adspec: received MTU 1500
  PATH sentto: localclient
  RESV rcvfrom: localclient
  Record route: 10.2.0.1 10.5.0.1 <self>

192.0.2.6
  From: 172.16.183.56, LSPstate: Up, ActiveRoute: 0
  LSPname: toPE2Primary192.0.2.6, LSPpath: Primary
  Suggested label received: -, Suggested label sent: -
  Recovery label received: -, Recovery label sent: -
  Resv style: 1 FF, Label in: 299936, Label out: 3
  Time left:  152, Since: Mon Jun 10 13:10:11 2013
  Tspec: rate 0bps size 0bps peak Infbps m 20 M 1500
  Port number: sender 2 receiver 59957 protocol 0
  Attrib flags: Non-PHP OOB
  PATH rcvfrom: 10.7.0.1 (ge-1/2/1.0) 106 pkts
  Adspec: received MTU 1500
  PATH sentto: localclient
  RESV rcvfrom: localclient
  Record route: 10.7.0.1 <self>
Total 2 displayed, Up 2, Down 0

Transit LSP: 0 sessions
Total 0 displayed, Up 0, Down 0
user@P1> show mpls lsp extensive
Ingress LSP: 0 sessions
Total 0 displayed, Up 0, Down 0

Egress LSP: 0 sessions
Total 0 displayed, Up 0, Down 0

Transit LSP: 3 sessions

192.0.2.6
  From: 172.16.183.55, LSPstate: Up, ActiveRoute: 0
  LSPname: toPE2Primary192.0.2.6, LSPpath: Primary
  Suggested label received: -, Suggested label sent: -
```

```
  Recovery label received: -, Recovery label sent: 299920
  Resv style: 1 FF, Label in: 299904, Label out: 299920
  Time left:  141, Since: Mon Jun 10 13:10:11 2013
  Tspec: rate 0bps size 0bps peak Infbps m 20 M 1500
  Port number: sender 2 receiver 17060 protocol 0
  Attrib flags: Non-PHP OOB
  PATH rcvfrom: 10.2.0.1 (ge-1/2/1.0) 106 pkts
  Adspec: received MTU 1500 sent MTU 1500
  PATH sentto: 10.5.0.2 (ge-1/2/2.0) 105 pkts
  RESV rcvfrom: 10.5.0.2 (ge-1/2/2.0) 105 pkts
  Explct route: 10.5.0.2 192.0.2.6 (link-id=2)
  Record route: 10.2.0.1 <self> 10.5.0.2

172.16.183.55
  From: 172.16.183.59, LSPstate: Up, ActiveRoute: 0
  LSPname: toPE1, LSPpath: Primary
  Suggested label received: -, Suggested label sent: -
  Recovery label received: -, Recovery label sent: 3
  Resv style: 1 FF, Label in: 299888, Label out: 3
  Time left:  158, Since: Mon Jun 10 13:10:11 2013
  Tspec: rate 0bps size 0bps peak Infbps m 20 M 1500
  Port number: sender 2 receiver 10941 protocol 0
  PATH rcvfrom: 10.5.0.2 (ge-1/2/2.0) 106 pkts
  Adspec: received MTU 1500 sent MTU 1500
  PATH sentto: 10.2.0.1 (ge-1/2/1.0) 105 pkts
  RESV rcvfrom: 10.2.0.1 (ge-1/2/1.0) 105 pkts
  Explct route: 10.2.0.1
  Record route: 10.5.0.2 <self> 10.2.0.1

172.16.183.55
  From: 172.16.183.56, LSPstate: Up, ActiveRoute: 0
  LSPname: toPE1, LSPpath: Primary
  Suggested label received: -, Suggested label sent: -
  Recovery label received: -, Recovery label sent: 3
  Resv style: 1 FF, Label in: 299920, Label out: 3
  Time left:  141, Since: Mon Jun 10 13:10:11 2013
  Tspec: rate 0bps size 0bps peak Infbps m 20 M 1500
  Port number: sender 2 receiver 59956 protocol 0
  PATH rcvfrom: 10.4.0.2 (ge-1/2/0.0) 105 pkts
  Adspec: received MTU 1500 sent MTU 1500
  PATH sentto: 10.2.0.1 (ge-1/2/1.0) 105 pkts
  RESV rcvfrom: 10.2.0.1 (ge-1/2/1.0) 105 pkts
  Explct route: 10.2.0.1
```

```
  Record route: 10.4.0.2 <self> 10.2.0.1
Total 3 displayed, Up 3, Down 0
```

Verifying BGP NRLI

Purpose

Check the details of the BGP VPN network layer reachability information.

Action

```
user@PE3> show bgp neighbor

Peer: 172.16.183.55+179 AS 64510 Local: 172.16.183.59+61747 AS 64510
  Type: Internal    State: Established    Flags: <Sync>
  Last State: OpenConfirm   Last Event: RecvKeepAlive
  Last Error: None
  Options: <Preference LocalAddress AddressFamily Rib-group Refresh>
  Address families configured: inet-vpn-unicast
  Local Address: 172.16.183.59 Holdtime: 90 Preference: 170
  NLRI configured with egress-protection: inet-vpn-unicast
  Egress-protection NLRI inet-vpn-unicast, keep-import: [ remote-vrf ]
  Number of flaps: 0
  Peer ID: 172.16.183.55   Local ID: 172.16.183.59     Active Holdtime: 90
  Keepalive Interval: 30         Group index: 0    Peer index: 0
  BFD: disabled, down
  NLRI for restart configured on peer: inet-vpn-unicast
  NLRI advertised by peer: inet-vpn-unicast
  NLRI for this session: inet-vpn-unicast
  Peer supports Refresh capability (2)
  Stale routes from peer are kept for: 300
  Peer does not support Restarter functionality
  NLRI that restart is negotiated for: inet-vpn-unicast
  NLRI of received end-of-rib markers: inet-vpn-unicast
  Peer supports 4 byte AS extension (peer-as 64510)
  Peer does not support Addpath
  Table bgp.l3vpn.0
    RIB State: BGP restart is complete
    RIB State: VPN restart is complete
    Send state: not advertising
    Active prefixes:              0
```

```
    Received prefixes:             0
    Accepted prefixes:             0
    Suppressed due to damping:     0
  Last traffic (seconds): Received 25   Sent 21   Checked 11
  Input messages:  Total 32046  Updates 7       Refreshes 0     Octets 609365
  Output messages: Total 32050  Updates 0       Refreshes 5     Octets 609010
  Output Queue[0]: 0

Peer: 172.16.183.56+62754 AS 64510 Local: 172.16.183.59+179 AS 64510
  Type: Internal    State: Established    Flags: <Sync>
  Last State: OpenConfirm   Last Event: RecvKeepAlive
  Last Error: None
  Options: <Preference LocalAddress AddressFamily Rib-group Refresh>
  Address families configured: inet-vpn-unicast
  Local Address: 172.16.183.59 Holdtime: 90 Preference: 170
  NLRI configured with egress-protection: inet-vpn-unicast
  Egress-protection NLRI inet-vpn-unicast, keep-import: [ remote-vrf ]
  Number of flaps: 1
  Last flap event: TransportError
  Peer ID: 172.16.183.56   Local ID: 172.16.183.59     Active Holdtime: 90
  Keepalive Interval: 30         Group index: 0    Peer index: 1
  BFD: disabled, down
  NLRI for restart configured on peer: inet-vpn-unicast
  NLRI advertised by peer: inet-vpn-unicast
  NLRI for this session: inet-vpn-unicast
  Peer supports Refresh capability (2)
  Stale routes from peer are kept for: 300
  Peer does not support Restarter functionality
  NLRI that restart is negotiated for: inet-vpn-unicast
  Peer supports 4 byte AS extension (peer-as 64510)
  Peer does not support Addpath
  Table bgp.l3vpn.0
    RIB State: BGP restart is complete
    RIB State: VPN restart is complete
    Send state: not advertising
    Active prefixes:               0
    Received prefixes:             0
    Accepted prefixes:             0
    Suppressed due to damping:     0
  Last traffic (seconds): Received 19   Sent 8    Checked 34
  Input messages:  Total 10025  Updates 0       Refreshes 2     Octets 190523
```

```
  Output messages: Total 10024  Updates 0       Refreshes 2     Octets 190504
  Output Queue[0]: 0
```

Meaning

`NLRI configured with egress-protection` shows the BGP family configured with egress protection. `egress-protection NLRI inet-vpn-unicast, keep-import: [remote-vrf]` shows the egress protection routing policy for the BGP group.

Verifying the Traffic Engineering Database

Purpose

On all devices, check the TED.

Action

```
user@PE1> show ted database

TED database: 9 ISIS nodes 5 INET nodes
ID                            Type Age(s) LnkIn LnkOut Protocol
P1.00(172.16.0.3)         Rtr      44     3      3 IS-IS(2)
    To: P1.02, Local: 10.2.0.2, Remote: 0.0.0.0
      Local interface index: 149, Remote interface index: 0
    To: PE2.02, Local: 10.4.0.1, Remote: 0.0.0.0
      Local interface index: 150, Remote interface index: 0
    To: PE3.03, Local: 10.5.0.1, Remote: 0.0.0.0
      Local interface index: 133, Remote interface index: 0
ID                            Type Age(s) LnkIn LnkOut Protocol
P1.02                     Net     111     2      2 IS-IS(2)
    To: PE1.00(172.16.183.55), Local: 0.0.0.0, Remote: 0.0.0.0
      Local interface index: 0, Remote interface index: 0
    To: P1.00(172.16.0.3), Local: 0.0.0.0, Remote: 0.0.0.0
      Local interface index: 0, Remote interface index: 0
ID                            Type Age(s) LnkIn LnkOut Protocol
PE2-192.0.2.6.00(192.0.2.6)   Rtr     345     2      2 IS-IS(2)
    To: PE2.00(172.16.183.56), Local: 0.0.0.0, Remote: 0.0.0.0
      Local interface index: 1, Remote interface index: 2147618817
    To: PE3.00(172.16.183.59), Local: 0.0.0.0, Remote: 0.0.0.0
      Local interface index: 2, Remote interface index: 2147618818
ID                            Type Age(s) LnkIn LnkOut Protocol
```

```
PE1.00(172.16.183.55)     Rtr     487     1      1 IS-IS(2)
    To: P1.02, Local: 10.2.0.1, Remote: 0.0.0.0
      Local interface index: 148, Remote interface index: 0
ID                            Type Age(s) LnkIn LnkOut Protocol
PE2.00(172.16.183.56)     Rtr     353     3      3 IS-IS(2)
    To: PE2.02, Local: 10.4.0.2, Remote: 0.0.0.0
      Local interface index: 155, Remote interface index: 0
    To: PE3.02, Local: 10.7.0.1, Remote: 0.0.0.0
      Local interface index: 153, Remote interface index: 0
    To: PE2-192.0.2.6.00(192.0.2.6), Local: 0.0.0.0, Remote: 0.0.0.0
      Local interface index: 2147618817, Remote interface index: 1
ID                            Type Age(s) LnkIn LnkOut Protocol
PE2.02                    Net      59     2      2 IS-IS(2)
    To: P1.00(172.16.0.3), Local: 0.0.0.0, Remote: 0.0.0.0
      Local interface index: 0, Remote interface index: 0
    To: PE2.00(172.16.183.56), Local: 0.0.0.0, Remote: 0.0.0.0
      Local interface index: 0, Remote interface index: 0
ID                            Type Age(s) LnkIn LnkOut Protocol
PE3.00(172.16.183.59)     Rtr     435     3      3 IS-IS(2)
    To: PE3.02, Local: 10.7.0.2, Remote: 0.0.0.0
      Local interface index: 154, Remote interface index: 0
    To: PE3.03, Local: 10.5.0.2, Remote: 0.0.0.0
      Local interface index: 158, Remote interface index: 0
    To: PE2-192.0.2.6.00(192.0.2.6), Local: 0.0.0.0, Remote: 0.0.0.0
      Local interface index: 2147618818, Remote interface index: 2
ID                            Type Age(s) LnkIn LnkOut Protocol
PE3.02                    Net     706     2      2 IS-IS(2)
    To: PE2.00(172.16.183.56), Local: 0.0.0.0, Remote: 0.0.0.0
      Local interface index: 0, Remote interface index: 0
    To: PE3.00(172.16.183.59), Local: 0.0.0.0, Remote: 0.0.0.0
      Local interface index: 0, Remote interface index: 0
ID                            Type Age(s) LnkIn LnkOut Protocol
PE3.03                    Net     583     2      2 IS-IS(2)
    To: P1.00(172.16.0.3), Local: 0.0.0.0, Remote: 0.0.0.0
      Local interface index: 0, Remote interface index: 0
    To: PE3.00(172.16.183.59), Local: 0.0.0.0, Remote: 0.0.0.0
      Local interface index: 0, Remote interface index: 0
user@PE2> show ted database
TED database: 9 ISIS nodes 5 INET nodes
ID                            Type Age(s) LnkIn LnkOut Protocol
P1.00(172.16.0.3)         Rtr      44     3      3 IS-IS(2)
    To: PE2.02, Local: 10.4.0.1, Remote: 0.0.0.0
      Local interface index: 150, Remote interface index: 0
```

```
    To: P1.02, Local: 10.2.0.2, Remote: 0.0.0.0
      Local interface index: 149, Remote interface index: 0
    To: PE3.03, Local: 10.5.0.1, Remote: 0.0.0.0
      Local interface index: 133, Remote interface index: 0
ID                             Type Age(s) LnkIn LnkOut Protocol
P1.02                      Net     111     2      2 IS-IS(2)
    To: P1.00(172.16.0.3), Local: 0.0.0.0, Remote: 0.0.0.0
      Local interface index: 0, Remote interface index: 0
    To: PE1.00(172.16.183.55), Local: 0.0.0.0, Remote: 0.0.0.0
      Local interface index: 0, Remote interface index: 0
ID                             Type Age(s) LnkIn LnkOut Protocol
PE2-192.0.2.6.00(192.0.2.6)    Rtr     345     2      2 IS-IS(2)
    To: PE2.00(172.16.183.56), Local: 0.0.0.0, Remote: 0.0.0.0
      Local interface index: 1, Remote interface index: 2147618817
    To: PE3.00(172.16.183.59), Local: 0.0.0.0, Remote: 0.0.0.0
      Local interface index: 2, Remote interface index: 2147618818
ID                             Type Age(s) LnkIn LnkOut Protocol
PE1.00(172.16.183.55)      Rtr     487     1      1 IS-IS(2)
    To: P1.02, Local: 10.2.0.1, Remote: 0.0.0.0
      Local interface index: 148, Remote interface index: 0
ID                             Type Age(s) LnkIn LnkOut Protocol
PE2.00(172.16.183.56)      Rtr     353     3      3 IS-IS(2)
    To: PE2.02, Local: 10.4.0.2, Remote: 0.0.0.0
      Local interface index: 155, Remote interface index: 0
    To: PE3.02, Local: 10.7.0.1, Remote: 0.0.0.0
      Local interface index: 153, Remote interface index: 0
    To: PE2-192.0.2.6.00(192.0.2.6), Local: 0.0.0.0, Remote: 0.0.0.0
      Local interface index: 2147618817, Remote interface index: 1
ID                             Type Age(s) LnkIn LnkOut Protocol
PE2.02                     Net      60     2      2 IS-IS(2)
    To: P1.00(172.16.0.3), Local: 0.0.0.0, Remote: 0.0.0.0
      Local interface index: 0, Remote interface index: 0
    To: PE2.00(172.16.183.56), Local: 0.0.0.0, Remote: 0.0.0.0
      Local interface index: 0, Remote interface index: 0
ID                             Type Age(s) LnkIn LnkOut Protocol
PE3.00(172.16.183.59)      Rtr     435     3      3 IS-IS(2)
    To: PE3.02, Local: 10.7.0.2, Remote: 0.0.0.0
      Local interface index: 154, Remote interface index: 0
    To: PE3.03, Local: 10.5.0.2, Remote: 0.0.0.0
      Local interface index: 158, Remote interface index: 0
    To: PE2-192.0.2.6.00(192.0.2.6), Local: 0.0.0.0, Remote: 0.0.0.0
      Local interface index: 2147618818, Remote interface index: 2
ID                             Type Age(s) LnkIn LnkOut Protocol
```

```
PE3.02                    Net     706     2      2 IS-IS(2)
    To: PE2.00(172.16.183.56), Local: 0.0.0.0, Remote: 0.0.0.0
      Local interface index: 0, Remote interface index: 0
    To: PE3.00(172.16.183.59), Local: 0.0.0.0, Remote: 0.0.0.0
      Local interface index: 0, Remote interface index: 0
ID                            Type Age(s) LnkIn LnkOut Protocol
PE3.03                    Net     583     2      2 IS-IS(2)
    To: P1.00(172.16.0.3), Local: 0.0.0.0, Remote: 0.0.0.0
      Local interface index: 0, Remote interface index: 0
    To: PE3.00(172.16.183.59), Local: 0.0.0.0, Remote: 0.0.0.0
      Local interface index: 0, Remote interface index: 0
user@PE3> show ted database
TED database: 9 ISIS nodes 5 INET nodes
ID                            Type Age(s) LnkIn LnkOut Protocol
P1.00(172.16.0.3)         Rtr      44     3      3 IS-IS(2)
    To: P1.02, Local: 10.2.0.2, Remote: 0.0.0.0
      Local interface index: 149, Remote interface index: 0
    To: PE2.02, Local: 10.4.0.1, Remote: 0.0.0.0
      Local interface index: 150, Remote interface index: 0
    To: PE3.03, Local: 10.5.0.1, Remote: 0.0.0.0
      Local interface index: 133, Remote interface index: 0
ID                            Type Age(s) LnkIn LnkOut Protocol
P1.02                     Net     111     2      2 IS-IS(2)
    To: PE1.00(172.16.183.55), Local: 0.0.0.0, Remote: 0.0.0.0
      Local interface index: 0, Remote interface index: 0
    To: P1.00(172.16.0.3), Local: 0.0.0.0, Remote: 0.0.0.0
      Local interface index: 0, Remote interface index: 0
ID                            Type Age(s) LnkIn LnkOut Protocol
PE2-192.0.2.6.00(192.0.2.6)   Rtr     345     2      2 IS-IS(2)
    To: PE2.00(172.16.183.56), Local: 0.0.0.0, Remote: 0.0.0.0
      Local interface index: 1, Remote interface index: 2147618817
    To: PE3.00(172.16.183.59), Local: 0.0.0.0, Remote: 0.0.0.0
      Local interface index: 2, Remote interface index: 2147618818
ID                            Type Age(s) LnkIn LnkOut Protocol
PE1.00(172.16.183.55)     Rtr     487     1      1 IS-IS(2)
    To: P1.02, Local: 10.2.0.1, Remote: 0.0.0.0
      Local interface index: 148, Remote interface index: 0
ID                            Type Age(s) LnkIn LnkOut Protocol
PE2.00(172.16.183.56)     Rtr     353     3      3 IS-IS(2)
    To: PE3.02, Local: 10.7.0.1, Remote: 0.0.0.0
      Local interface index: 153, Remote interface index: 0
    To: PE2.02, Local: 10.4.0.2, Remote: 0.0.0.0
      Local interface index: 155, Remote interface index: 0
```

```
    To: PE2-192.0.2.6.00(192.0.2.6), Local: 0.0.0.0, Remote: 0.0.0.0
      Local interface index: 2147618817, Remote interface index: 1
ID                            Type Age(s) LnkIn LnkOut Protocol
PE2.02                    Net      59     2      2 IS-IS(2)
    To: PE2.00(172.16.183.56), Local: 0.0.0.0, Remote: 0.0.0.0
      Local interface index: 0, Remote interface index: 0
    To: P1.00(172.16.0.3), Local: 0.0.0.0, Remote: 0.0.0.0
      Local interface index: 0, Remote interface index: 0
ID                            Type Age(s) LnkIn LnkOut Protocol
PE3.00(172.16.183.59)     Rtr     435     3      3 IS-IS(2)
    To: PE3.02, Local: 10.7.0.2, Remote: 0.0.0.0
      Local interface index: 154, Remote interface index: 0
    To: PE3.03, Local: 10.5.0.2, Remote: 0.0.0.0
      Local interface index: 158, Remote interface index: 0
    To: PE2-192.0.2.6.00(192.0.2.6), Local: 0.0.0.0, Remote: 0.0.0.0
      Local interface index: 2147618818, Remote interface index: 2
ID                            Type Age(s) LnkIn LnkOut Protocol
PE3.02                    Net     706     2      2 IS-IS(2)
    To: PE2.00(172.16.183.56), Local: 0.0.0.0, Remote: 0.0.0.0
      Local interface index: 0, Remote interface index: 0
    To: PE3.00(172.16.183.59), Local: 0.0.0.0, Remote: 0.0.0.0
      Local interface index: 0, Remote interface index: 0
ID                            Type Age(s) LnkIn LnkOut Protocol
PE3.03                    Net     583     2      2 IS-IS(2)
    To: PE3.00(172.16.183.59), Local: 0.0.0.0, Remote: 0.0.0.0
      Local interface index: 0, Remote interface index: 0
    To: P1.00(172.16.0.3), Local: 0.0.0.0, Remote: 0.0.0.0
      Local interface index: 0, Remote interface index: 0
-----


user@P1> show ted database
TED database: 9 ISIS nodes 5 INET nodes
ID                            Type Age(s) LnkIn LnkOut Protocol
P1.00(172.16.0.3)         Rtr      44     3      3 IS-IS(2)
    To: PE2.02, Local: 10.4.0.1, Remote: 0.0.0.0
      Local interface index: 150, Remote interface index: 0
    To: P1.02, Local: 10.2.0.2, Remote: 0.0.0.0
      Local interface index: 149, Remote interface index: 0
    To: PE3.03, Local: 10.5.0.1, Remote: 0.0.0.0
      Local interface index: 133, Remote interface index: 0
ID                            Type Age(s) LnkIn LnkOut Protocol
P1.02                     Net     111     2      2 IS-IS(2)
    To: P1.00(172.16.0.3), Local: 0.0.0.0, Remote: 0.0.0.0
```

```
      Local interface index: 0, Remote interface index: 0
    To: PE1.00(172.16.183.55), Local: 0.0.0.0, Remote: 0.0.0.0
      Local interface index: 0, Remote interface index: 0
ID                            Type Age(s) LnkIn LnkOut Protocol
PE2-192.0.2.6.00(192.0.2.6)   Rtr     345     2      2 IS-IS(2)
    To: PE2.00(172.16.183.56), Local: 0.0.0.0, Remote: 0.0.0.0
      Local interface index: 1, Remote interface index: 2147618817
    To: PE3.00(172.16.183.59), Local: 0.0.0.0, Remote: 0.0.0.0
      Local interface index: 2, Remote interface index: 2147618818
ID                            Type Age(s) LnkIn LnkOut Protocol
PE1.00(172.16.183.55)     Rtr     487     1      1 IS-IS(2)
    To: P1.02, Local: 10.2.0.1, Remote: 0.0.0.0
      Local interface index: 148, Remote interface index: 0
ID                            Type Age(s) LnkIn LnkOut Protocol
PE2.00(172.16.183.56)     Rtr     353     3      3 IS-IS(2)
    To: PE2.02, Local: 10.4.0.2, Remote: 0.0.0.0
      Local interface index: 155, Remote interface index: 0
    To: PE3.02, Local: 10.7.0.1, Remote: 0.0.0.0
      Local interface index: 153, Remote interface index: 0
    To: PE2-192.0.2.6.00(192.0.2.6), Local: 0.0.0.0, Remote: 0.0.0.0
      Local interface index: 2147618817, Remote interface index: 1
ID                            Type Age(s) LnkIn LnkOut Protocol
PE2.02                    Net      59     2      2 IS-IS(2)
    To: P1.00(172.16.0.3), Local: 0.0.0.0, Remote: 0.0.0.0
      Local interface index: 0, Remote interface index: 0
    To: PE2.00(172.16.183.56), Local: 0.0.0.0, Remote: 0.0.0.0
      Local interface index: 0, Remote interface index: 0
ID                            Type Age(s) LnkIn LnkOut Protocol
PE3.00(172.16.183.59)     Rtr     435     3      3 IS-IS(2)
    To: PE3.02, Local: 10.7.0.2, Remote: 0.0.0.0
      Local interface index: 154, Remote interface index: 0
    To: PE3.03, Local: 10.5.0.2, Remote: 0.0.0.0
      Local interface index: 158, Remote interface index: 0
    To: PE2-192.0.2.6.00(192.0.2.6), Local: 0.0.0.0, Remote: 0.0.0.0
      Local interface index: 2147618818, Remote interface index: 2
ID                            Type Age(s) LnkIn LnkOut Protocol
PE3.02                    Net     706     2      2 IS-IS(2)
    To: PE2.00(172.16.183.56), Local: 0.0.0.0, Remote: 0.0.0.0
      Local interface index: 0, Remote interface index: 0
    To: PE3.00(172.16.183.59), Local: 0.0.0.0, Remote: 0.0.0.0
      Local interface index: 0, Remote interface index: 0
ID                            Type Age(s) LnkIn LnkOut Protocol
PE3.03                    Net     583     2      2 IS-IS(2)
```

```
    To: P1.00(172.16.0.3), Local: 0.0.0.0, Remote: 0.0.0.0
      Local interface index: 0, Remote interface index: 0
    To: PE3.00(172.16.183.59), Local: 0.0.0.0, Remote: 0.0.0.0
      Local interface index: 0, Remote interface index: 0
```

Verifying the IS-IS Database

Purpose

On all devices, check the IS-IS database.

Action

```
user@PE1> show isis database

IS-IS level 1 link-state database:
  0 LSPs

IS-IS level 2 link-state database:
LSP ID                      Sequence Checksum Lifetime Attributes
P1.00-00                   0x46b   0x1924      590 L1 L2
P1.02-00                   0x465   0xe67a      523 L1 L2
PE2-192.0.2.6.00-00          0xd0e   0x6b8d     1086 L1 L2 Overload
PE1.00-00                  0x46f    0xa8b      992 L1 L2
PE2.00-00                  0x46b   0xefd6     1077 L1 L2
PE2.02-00                  0x464   0x4db4      573 L1 L2
PE3.00-00                  0x46f   0xb6e8     1016 L1 L2
PE3.02-00                  0x465   0x2675      762 L1 L2
PE3.03-00                  0x465   0x47b2      797 L1 L2
  9 LSPs

user@PE2> show isis database
IS-IS level 1 link-state database:
  0 LSPs

IS-IS level 2 link-state database:
LSP ID                      Sequence Checksum Lifetime Attributes
P1.00-00                   0x46b   0x1924      590 L1 L2
P1.02-00                   0x465   0xe67a      523 L1 L2
PE2-192.0.2.6.00-00          0xd0e   0x6b8d     1090 L1 L2 Overload
PE1.00-00                  0x46f    0xa8b      988 L1 L2
```

```
PE2.00-00                 0x46b   0xefd6     1080 L1 L2
PE2.02-00                 0x464   0x4db4      576 L1 L2
PE3.00-00                 0x46f   0xb6e8     1018 L1 L2
PE3.02-00                 0x465   0x2675      763 L1 L2
PE3.03-00                 0x465   0x47b2      799 L1 L2
  9 LSPs
user@PE3> show isis database
IS-IS level 1 link-state database:
  0 LSPs

IS-IS level 2 link-state database:
LSP ID                     Sequence Checksum Lifetime Attributes
P1.00-00                  0x46b   0x1924      590 L1 L2
P1.02-00                  0x465   0xe67a      523 L1 L2
PE2-192.0.2.6.00-00         0xd0e   0x6b8d     1088 L1 L2 Overload
PE1.00-00                 0x46f    0xa8b      988 L1 L2
PE2.00-00                 0x46b   0xefd6     1079 L1 L2
PE2.02-00                 0x464   0x4db4      575 L1 L2
PE3.00-00                 0x46f   0xb6e8     1020 L1 L2
PE3.02-00                 0x465   0x2675      765 L1 L2
PE3.03-00                 0x465   0x47b2      801 L1 L2
  9 LSPs
user@P1> show isis database
IS-IS level 1 link-state database:
  0 LSPs

IS-IS level 2 link-state database:
LSP ID                     Sequence Checksum Lifetime Attributes
P1.00-00                  0x46b   0x1924      592 L1 L2
P1.02-00                  0x465   0xe67a      525 L1 L2
PE2-192.0.2.6.00-00         0xd0e   0x6b8d     1088 L1 L2 Overload
PE1.00-00                 0x46f    0xa8b      990 L1 L2
PE2.00-00                 0x46b   0xefd6     1079 L1 L2
PE2.02-00                 0x464   0x4db4      575 L1 L2
PE3.00-00                 0x46f   0xb6e8     1018 L1 L2
PE3.02-00                 0x465   0x2675      763 L1 L2
PE3.03-00                 0x465   0x47b2      799 L1 L2
  9 LSPs
```

Provider Edge Link Protections in Layer 3 VPNs

IN THIS SECTION

This topic describes and provides examples on configuring precomputed protection path, which provides link protection and a backup path between a CE router and an alternative PE router.

Understanding Provider Edge Link Protection for BGP Labeled Unicast Paths

In MPLS service provider networks, when Layer 3 VPNs are used for carrier-of-carriers deployments, the protocol used to link the customer edge (CE) routers in one autonomous system (AS) and a provider edge (PE) router in another AS is BGP labeled-unicast. Reroute solutions between ASs are essential to help service providers ensure that network outages will have minimal impact on the data flows through the networks. A service provider that is a customer of another service provider can have different CE routers that are connected to the other service provider through different PE routers. This setup enables load balancing of traffic. However, this can lead to disruption in traffic if the link between one CE router and a PE router goes down. Therefore, a precomputed protection path should be configured such that if a link between a CE router and a PE router goes down, the protection path (also known as the backup path) between the other CE router and the alternative PE router can be used.

To configure a labeled-unicast path to be a protection path, use the `protection` statement at the `[edit routing-instances *instance-name* protocols bgp family inet labeled-unicast]` hierarchy level:

```
routing-instances {
    customer {
        instance-type vrf;
        ...
        protocols {
            bgp {
                family inet {
                    labeled-unicast {
```

```
                        protection;
                    }
                }
                family inet6 {
                    labeled-unicast {
                        protection;
                    }
                }
                type external;
                ...
            }
        }
    }
}
```

The `protection` statement indicates that protection is desired on prefixes received from the particular neighbor or family. After protection is enabled for a given family, group, or neighbor, protection entries are added for prefixes or next hops received from the given peer.

> **NOTE**: A protection path can be selected only if the best path has already been installed by BGP in the forwarding table. This is because a protection path cannot be used as the best path.
>
> To minimize packet loss when the protected path is down, also use the `per-prefix-label` statement at the `[edit routing-instances` *`instance-name`* `protocols bgp family inet labeled-unicast]` hierarchy level. Set this statement on every PE router within the AS containing the protected path.

The protection path selection takes place based on the value of two state flags:

- The `ProtectionPath` flag indicates paths desiring protection.
- The `ProtectionCand` flag indicates the route entry that can be used as a protection path.

> **NOTE**:
>
> - Provider edge link protection is configured only for external peers.
> - If provider edge link protection is configured with the `equal-external-internal` multipath statement, multipath takes precedence over protection.

Understanding Provider Edge Link Protection in Layer 3 VPNs

In an MPLS service provider network, a customer can have dual-homed CE routers that are connected to the service provider through different PE routers. This setup enables load balancing of traffic in the service provider network. However, this can lead to disruption in traffic if the link between a CE router and a PE router goes down. Hence, a precomputed protection path should be configured such that if a link between a CE router and a PE router goes down, the protection path (also known as the backup path) between the CE router and an alternate PE router can be used.

To configure a path to be a protection path, use the `protection` statement at the `[edit routing-instances` *`instance-name`* `protocols bgp family inet unicast]` hierarchy level:

```
routing-instances {
    customer {
        instance-type vrf;
        ...
        protocols {
            bgp {
                type external;
                ...
                family inet {
                    unicast {
                        protection;
                    }
                }
                family inet6 {
                    unicast {
                        protection;
                    }
                }
            }
        }
    }
}
```

The `protection` statement indicates that protection is required on prefixes received from the particular neighbor or family. After protection is enabled for a given family, group, or neighbor, protection entries are added for prefixes or next hops received from the given peer.

NOTE: A protection path can be selected only if the best path has already been installed by BGP in the forwarding table. This is because a protection path cannot be used as the best path.

NOTE: The option `vrf-table-label` must be configured under the `[routing-instances` *`instance-name`*`]` hierarchy for the routers that have protected PE-CE links. This applies to Junos OS Releases 12.3 through 13.2 inclusive.

The protection path selection takes place based on the value of two state flags:

- The `ProtectionPath` flag indicates paths requesting protection.
- The `ProtectionCand` flag indicates the route entry that can be used as a protection path.

NOTE:

- Provider edge link protection is configured only for external peers.
- If provider edge link protection is configured with the `equal-external-internal` multipath statement, multipath takes precedence over protection.

Example: Configuring Provider Edge Link Protection in Layer 3 VPNs

IN THIS SECTION

This example shows how to configure a provider edge protection path that can be used in case of a link failure in an MPLS network.

Requirements

This example uses the following hardware components, software components and configuration options:

- M Series Multiservice Edge Routers, MX Series 5G Universal Routing Platforms, or T Series Core Routers
- Junos OS Release 12.3 through 13.2 inclusive
- The option `vrf-table-label` must be enabled at the `[routing-instances *instance-name*]` hierarchy level for routers with protected PE-CE links.

Overview

IN THIS SECTION

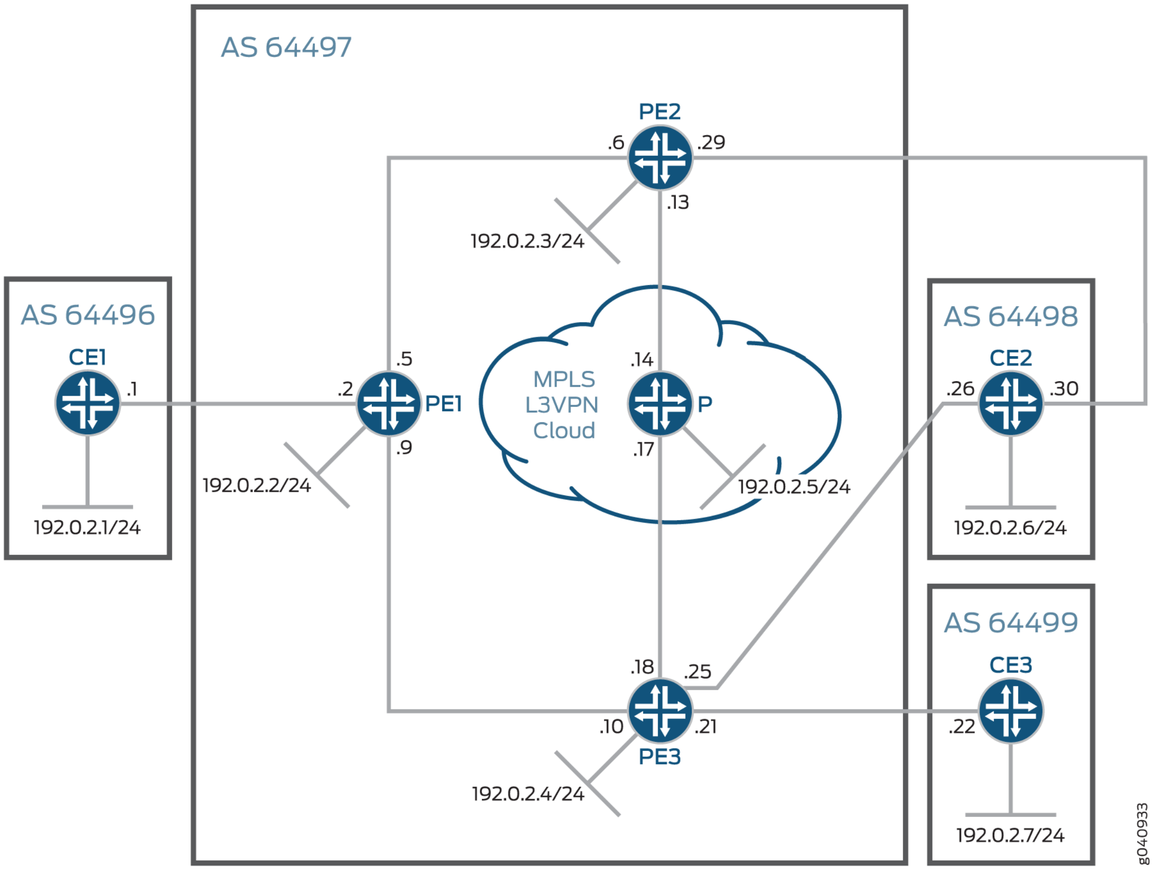
The following example shows how to configure provider edge link protection in a Layer 3 VPN.

Topology

In this example, a Layer 3 VPN is set up by configuring three customer edge devices and three service provider edge devices in four autonomous systems. The CE devices are configured in AS 64496, AS 64498, and AS 64499. The PE devices are configured in AS 64497.

Figure 83 on page 1070 shows the topology used in this example.

Figure 83: Provider Edge Link Protection in a Layer 3 VPN



The aim of this example is to protect the provider edge link between Routers PE3 and CE2. You configure protection on the primary link between Routers PE3 and CE2 by routing traffic through the backup link of PE2-CE2 when the PE3-CE2 link goes down.

Configuration

IN THIS SECTION

CLI Quick Configuration

To quickly configure this example, copy the following commands, paste them into a text file, remove any line breaks, change any details necessary to match your network configuration, and then copy and paste the commands into the CLI at the `[edit]` hierarchy level.

Router CE1

```
set interfaces ge-2/0/0 unit 0 description toPE1
set interfaces ge-2/0/0 unit 0 family inet address 10.1.1.1/30
set interfaces ge-2/0/0 unit 0 family inet6 address 2001:db8:0:1::/64 eui-64
set interfaces ge-2/0/0 unit 0 family mpls
set interfaces lo0 unit 0 family inet address 192.0.2.1/24
set interfaces lo0 unit 0 family inet6 address 2001:db8::1/128
set routing-options router-id 192.0.2.1
set routing-options autonomous-system 64496
set protocols bgp group toPE1 type external
set protocols bgp group toPE1 export send-direct
set protocols bgp group toPE1 peer-as 64497
set protocols bgp group toPE1 neighbor 10.1.1.2
set policy-options policy-statement send-direct from protocol direct
set policy-options policy-statement send-direct then accept
```

Router PE1

```
set interfaces ge-2/0/0 unit 0 description toCE1
set interfaces ge-2/0/0 unit 0 family inet address 10.1.1.2/30
set interfaces ge-2/0/0 unit 0 family inet6 address 2001:db8:0:1::/64 eui-64
set interfaces ge-2/0/0 unit 0 family mpls
set interfaces ge-2/0/1 unit 0 description toPE2
set interfaces ge-2/0/1 unit 0 family inet address 10.1.1.5/30
set interfaces ge-2/0/1 unit 0 family inet6 address 2001:db8:0:5::/64 eui-64
set interfaces ge-2/0/1 unit 0 family mpls
set interfaces ge-2/0/2 unit 0 description toPE3
set interfaces ge-2/0/2 unit 0 family inet address 10.1.1.9/30
set interfaces ge-2/0/2 unit 0 family inet6 address 2001:db8:0:9::/64 eui-64
set interfaces ge-2/0/2 unit 0 family mpls
set interfaces lo0 unit 0 family inet address 192.0.2.2/24
set interfaces lo0 unit 0 family inet6 address 2001:db8::2/128
set protocols mpls interface all
set protocols ldp interface all
set protocols ospf area 0.0.0.0 interface lo0.0 passive
```

```
set protocols ospf area 0.0.0.0 interface ge-2/0/1.0 metric 10
set protocols ospf area 0.0.0.0 interface ge-2/0/2.0 metric 10
set protocols ospf3 area 0.0.0.0 interface lo0.0 passive
set protocols ospf3 area 0.0.0.0 interface ge-2/0/1.0 metric 10
set protocols ospf3 area 0.0.0.0 interface ge-2/0/2.0 metric 10
set protocols bgp group toInternal type internal
set protocols bgp group toInternal family inet-vpn unicast
set protocols bgp group toInternal family inet6-vpn unicast
set protocols bgp group toInternal multipath
set protocols bgp group toInternal local-address 192.0.2.2
set protocols bgp group toInternal neighbor 192.0.2.3
set protocols bgp group toInternal neighbor 192.0.2.4
set routing-options router-id 192.0.2.2
set routing-options autonomous-system 64497
set routing-options forwarding-table export lb
set routing-instances radium instance-type vrf
set routing-instances radium interface ge-2/0/0.0
set routing-instances radium route-distinguisher 64497:1
set routing-instances radium vrf-target target:64497:1
set routing-instances radium protocols bgp group toCE1 type external
set routing-instances radium protocols bgp group toCE1 peer-as 64496
set routing-instances radium protocols bgp group toCE1 neighbor 10.1.1.1
set policy-options policy-statement lb then load-balance per-packet
```

Router PE2

```
set interfaces ge-2/0/0 unit 0 description toPE1
set interfaces ge-2/0/0 unit 0 family inet address 10.1.1.6/30
set interfaces ge-2/0/0 unit 0 family inet6 address 2001:db8:0:5::/64 eui-64
set interfaces ge-2/0/0 unit 0 family mpls
set interfaces ge-2/0/1 unit 0 description toP
set interfaces ge-2/0/1 unit 0 family inet address 10.1.1.13/30
set interfaces ge-2/0/1 unit 0 family inet6 address 2001:db8:0:13::/64 eui-64
set interfaces ge-2/0/1 unit 0 family mpls
set interfaces ge-2/0/2 unit 0 description toCE2
set interfaces ge-2/0/2 unit 0 family inet address 10.1.1.29/30
set interfaces ge-2/0/2 unit 0 family inet6 address 2001:db8:0:29::/64 eui-64
set interfaces ge-2/0/2 unit 0 family mpls
set interfaces lo0 unit 0 family inet address 192.0.2.3/24
set interfaces lo0 unit 0 family inet6 address 2001:db8::3/128
set protocols mpls interface all
set protocols ldp interface all
```

```
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set protocols ospf area 0.0.0.0 interface ge-2/0/0.0 metric 10
set protocols ospf area 0.0.0.0 interface ge-2/0/1.0 metric 5
set protocols ospf3 area 0.0.0.0 interface lo0.0 passive
set protocols ospf3 area 0.0.0.0 interface ge-2/0/0.0 metric 10
set protocols ospf3 area 0.0.0.0 interface ge-2/0/1.0 metric 5
set protocols bgp group toInternal type internal
set protocols bgp group toInternal family inet-vpn unicast
set protocols bgp group toInternal family inet6-vpn unicast
set protocols bgp group toInternal multipath
set protocols bgp group toInternal local-address 192.0.2.3
set protocols bgp group toInternal neighbor 192.0.2.2
set protocols bgp group toInternal neighbor 192.0.2.4
set routing-options router-id 192.0.2.3
set routing-options autonomous-system 64497
set routing-options forwarding-table export lb
set routing-instances radium instance-type vrf
set routing-instances radium interface ge-2/0/2.0
set routing-instances radium route-distinguisher 64497:1
set routing-instances radium vrf-target target:64497:1
set routing-instances radium protocols bgp group toCE2 type external
set routing-instances radium protocols bgp group toCE2 peer-as 64498
set routing-instances radium protocols bgp group toCE2 neighbor 10.1.1.30
set policy-options policy-statement lb then load-balance per-packet
```

Router PE3

```
set interfaces ge-2/0/0 unit 0 description toPE1
set interfaces ge-2/0/0 unit 0 family inet address 10.1.1.10/30
set interfaces ge-2/0/0 unit 0 family inet6 address 2001:db8:0:9::/64 eui-64
set interfaces ge-2/0/0 unit 0 family mpls
set interfaces ge-2/0/1 unit 0 description toP
set interfaces ge-2/0/1 unit 0 family inet address 10.1.1.18/30
set interfaces ge-2/0/1 unit 0 family inet6 address 2001:db8:0:17::/64 eui-64
set interfaces ge-2/0/1 unit 0 family mpls
set interfaces ge-2/0/2 unit 0 description toCE2
set interfaces ge-2/0/2 unit 0 family inet address 10.1.1.25/30
set interfaces ge-2/0/2 unit 0 family inet6 address 2001:db8:0:25::/64 eui-64
set interfaces ge-2/0/2 unit 0 family mpls
set interfaces ge-2/0/3 unit 0 description toCE3
set interfaces ge-2/0/3 unit 0 family inet address 10.1.1.21/30
set interfaces ge-2/0/3 unit 0 family inet6 address 2001:db8:0:21::/64 eui-64
```

```
set interfaces ge-2/0/3 unit 0 family mpls
set interfaces lo0 unit 0 family inet address 192.0.2.4/24
set interfaces lo0 unit 0 family inet6 address 2001:db8::4/128
set protocols mpls interface all
set protocols ldp interface all
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set protocols ospf area 0.0.0.0 interface ge-2/0/1.0 metric 5
set protocols ospf area 0.0.0.0 interface ge-2/0/0.0 metric 10
set protocols ospf3 area 0.0.0.0 interface lo0.0 passive
set protocols ospf3 area 0.0.0.0 interface ge-2/0/1.0 metric 5
set protocols ospf3 area 0.0.0.0 interface ge-2/0/0.0 metric 10
set protocols bgp group toInternal type internal
set protocols bgp group toInternal family inet-vpn unicast
set protocols bgp group toInternal family inet6-vpn unicast
set protocols bgp group toInternal multipath
set protocols bgp group toInternal local-address 192.0.2.4
set protocols bgp group toInternal neighbor 192.0.2.2
set protocols bgp group toInternal neighbor 192.0.2.3
set routing-options router-id 192.0.2.4
set routing-options autonomous-system 64497
set routing-options forwarding-table export lb
set routing-instances radium instance-type vrf
set routing-instances radium vrf-table-label
set routing-instances radium interface ge-2/0/2.0
set routing-instances radium interface ge-2/0/3.0
set routing-instances radium route-distinguisher 64497:1
set routing-instances radium vrf-target target:64497:1
set routing-instances radium protocols bgp group toCE2 type external
set routing-instances radium protocols bgp group toCE2 peer-as 64498
set routing-instances radium protocols bgp group toCE2 neighbor 10.1.1.26
set routing-instances radium protocols bgp group toCE2 family inet unicast protection
set routing-instances radium protocols bgp group toCE2 family inet6 unicast protection
set routing-instances radium protocols bgp group toCE3 type external
set routing-instances radium protocols bgp group toCE3 peer-as 64499
set routing-instances radium protocols bgp group toCE3 neighbor 10.1.1.22
set policy-options policy-statement lb then load-balance per-packet
```

Router P

```
set interfaces ge-2/0/0 unit 0 description toPE2
set interfaces ge-2/0/0 unit 0 family inet address 10.1.1.14/30
set interfaces ge-2/0/0 unit 0 family inet6 address 2001:db8:0:13::/64 eui-64
```

```
set interfaces ge-2/0/0 unit 0 family mpls
set interfaces ge-2/0/1 unit 0 description toPE3
set interfaces ge-2/0/1 unit 0 family inet address 10.1.1.17/30
set interfaces ge-2/0/1 unit 0 family inet6 address 2001:db8:0:17::/64 eui-64
set interfaces ge-2/0/1 unit 0 family mpls
set interfaces lo0 unit 0 family inet address 192.0.2.5/24
set interfaces lo0 unit 0 family inet6 address 2001:db8::5/128
set routing-options router-id 192.0.2.5
set routing-options autonomous-system 64497
set protocols mpls interface all
set protocols ldp interface all
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set protocols ospf area 0.0.0.0 interface ge-2/0/0.0 metric 5
set protocols ospf area 0.0.0.0 interface ge-2/0/1.0 metric 5
set protocols ospf3 area 0.0.0.0 interface lo0.0 passive
set protocols ospf3 area 0.0.0.0 interface ge-2/0/0.0 metric 5
set protocols ospf3 area 0.0.0.0 interface ge-2/0/1.0 metric 5
```

Router CE2

```
set interfaces ge-2/0/0 unit 0 description toPE2
set interfaces ge-2/0/0 unit 0 family inet address 10.1.1.30/30
set interfaces ge-2/0/0 unit 0 family inet6 address 2001:db8:0:29::/64 eui-64
set interfaces ge-2/0/0 unit 0 family mpls
set interfaces ge-2/0/1 unit 0 description toPE3
set interfaces ge-2/0/1 unit 0 family inet address 10.1.1.26/30
set interfaces ge-2/0/1 unit 0 family inet6 address 2001:db8:0:25::/64 eui-64
set interfaces ge-2/0/1 unit 0 family mpls
set interfaces lo0 unit 0 family inet address 192.0.2.6/24
set interfaces lo0 unit 0 family inet6 address 2001:db8::6/128
set routing-options router-id 192.0.2.6
set routing-options autonomous-system 64498
set protocols bgp group toAS2 type external
set protocols bgp group toAS2 export send-direct
set protocols bgp group toAS2 peer-as 64497
set protocols bgp group toAS2 neighbor 10.1.1.25
set protocols bgp group toAS2 neighbor 10.1.1.29
set policy-options policy-statement send-direct from protocol direct
set policy-options policy-statement send-direct then accept
```

Router CE3

```
set interfaces ge-2/0/0 unit 0 description toPE3
set interfaces ge-2/0/0 unit 0 family inet address 10.1.1.22/30
set interfaces ge-2/0/0 unit 0 family inet6 address 2001:db8:0:21::/64 eui-64
set interfaces ge-2/0/0 unit 0 family mpls
set interfaces lo0 unit 0 family inet address 192.0.2.7/24
set interfaces lo0 unit 0 family inet6 address 2001:db8::7/128
set routing-options router-id 192.0.2.7
set routing-options autonomous-system 64499
set protocols bgp group toPE3 type external
set protocols bgp group toPE3 export send-direct
set protocols bgp group toPE3 peer-as 64497
set protocols bgp group toPE3 neighbor 10.1.1.21
set policy-options policy-statement send-direct from protocol direct
set policy-options policy-statement send-direct then accept
```

Configuring Provider Edge Link Protection in Layer 3 VPNs

Step-by-Step Procedure

The following example requires that you navigate various levels in the configuration hierarchy. For information about navigating the CLI, see *Using the CLI Editor in Configuration Mode* in the CLI User Guide.

To configure provider edge link protection:

1. Configure the router interfaces.

   ```
   [edit interfaces]
   user@PE3# set ge-2/0/0 unit 0 description toPE1
   user@PE3# set ge-2/0/0 unit 0 family inet address 10.1.1.10/30
   user@PE3# set ge-2/0/0 unit 0 family inet6 address 2001:db8:0:9::/64 eui-64
   user@PE3# set ge-2/0/0 unit 0 family mpls
   ```

   ```
   user@PE3# set ge-2/0/1 unit 0 description toP
   user@PE3# set ge-2/0/1 unit 0 family inet address 10.1.1.18/30
   ```

```
user@PE3# set ge-2/0/1 unit 0 family inet6 address 2001:db8:0:17::/64 eui-64
user@PE3# set ge-2/0/1 unit 0 family mpls
```

```
user@PE3# set ge-2/0/2 unit 0 description toCE2
user@PE3# set ge-2/0/2 unit 0 family inet address 10.1.1.25/30
user@PE3# set ge-2/0/2 unit 0 family inet6 address 12001:db8:0:25::/64 eui-64
user@PE3# set ge-2/0/2 unit 0 family mpls
```

```
user@PE3# set ge-2/0/3 unit 0 description toCE3
user@PE3# set ge-2/0/3 unit 0 family inet address 10.1.1.21/30
user@PE3# set ge-2/0/3 unit 0 family inet6 address 2001:db8:0:21::/64 eui-64
user@PE3# set ge-2/0/3 unit 0 family mpls
```

```
user@PE3# set lo0 unit 0 family inet address 192.0.2.4/24
user@PE3# set lo0 unit 0 family inet6 address 2001:db8::4/128
```

Similarly, configure the interfaces on all other routers.

2. Configure the router ID and autonomous system (AS) number.

```
[edit routing-options]
user@PE3# set  router-id 192.0.2.4
user@PE3# set  autonomous-system 64497
```

Similarly, configure the router ID and AS number for all other routers. In this example, the router ID is chosen to be identical to the loopback address configured on the router.

3. Configure MPLS and LDP on all interfaces of Router PE3.

```
[edit protocols]
user@PE3# set mpls interface all
user@PE3# set ldp interface all
```

Similarly, configure other PE routers.

4. Configure an IGP on the core-facing interfaces of Router PE3.

   ```
   [edit protocols ospf area 0.0.0.0]
   user@PE3# set interface lo0.0 passive
   user@PE3# set interface ge-2/0/1.0 metric 5
   user@PE3# set interface ge-2/0/0.0 metric 10
   ```

   ```
   [edit protocols ospf3 area 0.0.0.0]
   user@PE3# set interface lo0.0 passive
   user@PE3# set interface ge-2/0/1.0 metric 5
   user@PE3# set interface ge-2/0/0.0 metric 10
   ```

   Similarly, configure other PE routers.

5. Configure a policy that exports the routes from the routing table into the forwarding table on Router PE3.

   ```
   [edit policy-options]
   user@PE3# set policy-statement lb then load-balance per-packet
   ```

   ```
   [edit routing-options]
   user@PE3# set forwarding-table export lb
   ```

   Similarly, configure other PE routers.

6. Configure BGP on Router CE2, and include a policy for exporting routes to and from the service provider network.

   ```
   [edit policy-options]
   user@CE2# set policy-statement send-direct from protocol direct
   user@CE2# set policy-statement send-direct then accept
   ```

   ```
   [edit protocols bgp group toAS2]
   user@CE2# set type external
   user@CE2# set export send-direct
   user@CE2# set peer-as 64497
   ```

```
user@CE2# set neighbor 10.1.1.25
user@CE2# set neighbor 10.1.1.29
```

Similarly, configure other CE routers.

7. Configure BGP on Router PE3 for routing within the provider core.

```
[edit protocols bgp group toInternal]
user@PE3# set type internal
user@PE3# set family inet-vpn unicast
user@PE3# set family inet6-vpn unicast
user@PE3# set multipath
user@PE3# set local-address 192.0.2.4
user@PE3# set neighbor 192.0.2.2
user@PE3# set neighbor 192.0.2.3
```

Similarly, configure other PE routers.

8. Configure the Layer 3 VPN routing instance on Router PE3.

```
[set routing-instances radium]
user@PE3# set instance-type vrf
user@PE3# set vrf-table-label
user@PE3# set interface ge-2/0/2.0
user@PE3# set interface ge-2/0/3.0
user@PE3# set route-distinguisher 64497:1
user@PE3# set vrf-target target:64497:1
```

```
[edit routing-instances radium protocols bgp group toCE2]
user@PE3# set type external
user@PE3# set peer-as 64498
user@PE3# set neighbor 10.1.1.26
```

```
[edit routing-instances radium protocols bgp group toCE3]
user@PE3# set type external
user@PE3# set peer-as 64499
user@PE3# set neighbor 10.1.1.22
```

Similarly, configure other PE routers.

9. Configure provider edge link protection on the link between Routers PE3 and CE2.

```
[edit routing-instances radium protocols bgp group toCE2]
user@PE3# set family inet unicast protection
user@PE3# set family inet6 unicast protection
```

Results

From configuration mode, confirm your configuration by entering the `show interfaces`, `show routing-options`, `show policy-options`, `show protocols` , and `show routing-instances` commands.

If the output does not display the intended configuration, repeat the instructions in this example to correct the configuration.

```
user@PE3# show interfaces
ge-2/0/0 {
    unit 0 {
        description toPE1;
        family inet {
            address 10.1.1.10/30;
        }
        family inet6 {
            address 2001:db8:0:9::/64 {
                eui-64;
            }
        }
        family mpls;
    }
}

ge-2/0/1 {
    unit 0 {
        description toP;
        family inet {
            address 10.1.1.18/30;
        }
        family inet6 {
            address 2001:db8:0:17::/64 {
                eui-64;
            }
        }
```

```
        family mpls;
    }
}

ge-2/0/2 {
    unit 0 {
        description toCE2;
        family inet {
            address 10.1.1.25/30;
        }
        family inet6 {
            address 2001:db8:0:25::/64 {
                eui-64;
            }
        }
        family mpls;
    }
}

ge-2/0/3 {
    unit 0 {
        description toCE3;
         family inet {
            address 10.1.1.21/30;
        }
        family inet6 {
            address 2001:db8:0:21::/64 {
                eui-64;
            }
        }
        family mpls;
    }
}

lo0 {
    unit 0 {
        family inet {
            address 192.0.2.4/24;
        }
        family inet6 {
            address 2001:db8::4/128;
        }
```

```
    }
}
```

```
user@PE3# show routing-options
router-id 192.0.2.4;
autonomous-system 64497;
forwarding-table {
    export lb;
}
```

```
user@PE3# show policy-options
policy-statement lb {
        then {
            load-balance per-packet;
        }
    }
```

```
user@PE3# show protocols
mpls {
    interface all;
}
bgp {
    group toInternal {
        type internal;
        local-address 192.0.2.4;
        family inet-vpn {
            unicast;
        }
        family inet6-vpn {
            unicast;
        }
        multipath;
        neighbor 192.0.2.2;
        neighbor 192.0.2.3;
    }
}
ospf {
    area 0.0.0.0 {
        interface lo0.0 {
```

```
            passive;
        }
        interface ge-2/0/1.0 {
            metric 5;
        }
        interface ge-2/0/0.0 {
            metric 10;
        }
    }
}
ospf3 {
    area 0.0.0.0 {
        interface lo0.0 {
            passive;
        }
        interface ge-2/0/1.0 {
            metric 5;
        }
        interface ge-2/0/0.0 {
            metric 10;
        }
    }
}
ldp {
    interface all;
}
```

```
user@PE3# show routing-instances
radium {
    instance-type vrf;
    interface ge-2/0/2.0;
    interface ge-2/0/3.0;
    route-distinguisher 64497:1;
    vrf-target target:64497:1;
    protocols {
        bgp {
            group toCE2 {
                type external;
                family inet {
                    unicast {
                        protection;
```

```
                    }
                }
                family inet6 {
                    unicast {
                        protection;
                    }
                }
                peer-as 64498;
                neighbor 10.1.1.26;
            }
            group toCE3 {
                type external;
                peer-as 64499;
                neighbor 10.1.1.22;
            }
        }
    }
}
```

Run these commands on all other routers to confirm the configurations. If you are done configuring the routers, enter `commit` from configuration mode.

Verification

IN THIS SECTION

Confirm that the configuration is working properly.

Verifying BGP

Purpose

Verify that BGP is functional in the Layer 3 VPN.

Action

From operational mode on Router PE3, run the `show route protocol bgp` command.

```
user@PE3> show route protocol bgp
inet.0: 11 destinations, 11 routes (11 active, 0 holddown, 0 hidden)

inet.3: 3 destinations, 3 routes (3 active, 0 holddown, 0 hidden)

radium.inet.0: 9 destinations, 14 routes (9 active, 0 holddown, 0 hidden)
@ = Routing Use Only, # = Forwarding Use Only
+ = Active Route, - = Last Active, * = Both

192.0.2.1/24         *[BGP/170] 00:09:15, localpref 100, from 192.0.2.2
                      AS path: 64496 I, validation-state: unverified
                    > to 10.1.1.9 via ge-2/0/0.0, Push 299792
192.0.2.6/24         @[BGP/170] 00:09:40, localpref 100
                      AS path: 64498 I, validation-state: unverified
                    > to 10.1.1.26 via ge-2/0/2.0
                    [BGP/170] 00:09:07, localpref 100, from 192.0.2.3
                      AS path: 64498 I, validation-state: unverified
                    > to 10.1.1.17 via ge-2/0/1.0, Push 299792, Push 299776(top)
192.0.2.7/24         *[BGP/170] 00:09:26, localpref 100
                      AS path: 64499 I, validation-state: unverified
                    > to 10.1.1.22 via ge-2/0/3.0
10.1.1.0/30        *[BGP/170] 00:09:15, localpref 100, from 192.0.2.2
                      AS path: I, validation-state: unverified
                    > to 10.1.1.9 via ge-2/0/0.0, Push 299792
10.1.1.20/30        [BGP/170] 00:09:26, localpref 100
                      AS path: 64499 I, validation-state: unverified
                    > to 10.1.1.22 via ge-2/0/3.0
10.1.1.24/30        [BGP/170] 00:09:40, localpref 100
                      AS path: 64498 I, validation-state: unverified
                    > to 10.1.1.26 via ge-2/0/2.0
10.1.1.28/30       *[BGP/170] 00:09:07, localpref 100, from 192.0.2.3
                      AS path: I, validation-state: unverified
                    > to 10.1.1.17 via ge-2/0/1.0, Push 299792, Push 299776(top)
                    [BGP/170] 00:09:40, localpref 100
                      AS path: 64498 I, validation-state: unverified
                    > to 10.1.1.26 via ge-2/0/2.0

mpls.0: 11 destinations, 11 routes (11 active, 0 holddown, 0 hidden)
```

```
bgp.l3vpn.0: 4 destinations, 4 routes (4 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

64497:1:192.0.2.1/24
                   *[BGP/170] 00:09:15, localpref 100, from 192.0.2.2
                      AS path: 64496 I, validation-state: unverified
                    > to 10.1.1.9 via ge-2/0/0.0, Push 299792
64497:1:192.0.2.6/24
                   *[BGP/170] 00:09:07, localpref 100, from 192.0.2.3
                      AS path: 64498 I, validation-state: unverified
                    > to 10.1.1.17 via ge-2/0/1.0, Push 299792, Push 299776(top)
64497:1:10.1.1.0/30
                   *[BGP/170] 00:09:15, localpref 100, from 192.0.2.2
                      AS path: I, validation-state: unverified
                    > to 10.1.1.9 via ge-2/0/0.0, Push 299792
64497:1:10.1.1.28/30
                   *[BGP/170] 00:09:07, localpref 100, from 192.0.2.3
                      AS path: I, validation-state: unverified
                    > to 10.1.1.17 via ge-2/0/1.0, Push 299792, Push 299776(top)

inet6.0: 12 destinations, 13 routes (12 active, 0 holddown, 0 hidden)

radium.inet6.0: 7 destinations, 8 routes (7 active, 0 holddown, 0 hidden)
```

The output shows all the BGP routes in the routing table of Router PE3. This indicates that BGP is functioning as required.

Similarly, run this command on other routers to check if BGP is operational.

Meaning

BGP is functional in the Layer 3 VPN.

Verifying Provider Edge Link Protection

Purpose

Verify that the provider edge link between Routers PE2 and CE2 is protected.

Action

To verify that provider edge link protection is configured correctly:

1. Confirm that a route on Router CE2 is advertised to Router PE3, directly and through Router PE2.

   If the route is advertised correctly, you will see multiple paths for the route.

   From operational mode on Router PE3, run the `show route` *`destination-prefix`* command.

   ```
   user@PE3> show route 192.0.2.6
   radium.inet.0: 9 destinations, 14 routes (9 active, 0 holddown, 0 hidden)
   @ = Routing Use Only, # = Forwarding Use Only
   + = Active Route, - = Last Active, * = Both

   192.0.2.6/24         @[BGP/170] 02:55:36, localpref 100
                          AS path: 64498 I, validation-state: unverified
                        > to 10.1.1.26 via ge-2/0/2.0
                        [BGP/170] 00:10:13, localpref 100, from 192.0.2.3
                          AS path: 64498 I, validation-state: unverified
                        > to 10.1.1.17 via ge-2/0/1.0, Push 299840, Push 299776(top)
                       #[Multipath/255] 00:10:13
                        > to 10.1.1.26 via ge-2/0/2.0
                          to 10.1.1.17 via ge-2/0/1.0, Push 299840, Push 299776(top)
   ```

   The output verifies the presence of multiple paths from Router PE3 to the destination route, **192.0.2.6**, on Router CE2. The first path is directly through the PE3-CE2 link (**10.1.1.26**). The second path is through the provider core and PE2 (**10.1.1.17**).

2. Verify that the protection path is correctly configured by confirming that the weight for the active path being protected is `0x1`, and the weight for the protection candidate path is `0x4000`.

   From operational mode on Router PE3, run the `show route` *`destination-prefix`* `extensive` command.

   ```
   user@PE3> show route 192.0.2.6 extensive
   radium.inet.0: 9 destinations, 14 routes (9 active, 0 holddown, 0 hidden)
   192.0.2.6/24 (3 entries, 2 announced)
           State: <CalcForwarding>
   TSI:
   KRT in-kernel 192.0.2.6/24 -> {list:10.1.1.26, indirect(1048584)}
   Page 0 idx 1 Type 1 val 9229c38
       Nexthop: Self
       AS path: [64497] 64498 I
   ```

```
    Communities:
Page 0 idx 2 Type 1 val 9229cc4
    Flags: Nexthop Change
    Nexthop: Self
    Localpref: 100
    AS path: [64497] 64498 I
    Communities: target:64497:1
Path 192.0.2.6 from 10.1.1.26 Vector len 4.  Val: 1 2
        @BGP    Preference: 170/-101
                Next hop type: Router, Next hop index: 994
                Address: 0x9240a74
                Next-hop reference count: 5
                Source: 10.1.1.26
              Next hop: 10.1.1.26 via ge-2/0/2.0, selected
                Session Id: 0x200001
                State: <Active Ext ProtectionPath ProtectionCand>
                Peer AS: 64498
                Age: 2:55:54
                Validation State: unverified
                Task: BGP_64498.10.1.1.26+52214
                Announcement bits (1): 2-BGP_RT_Background
                AS path: 64498 I
                Accepted
                Localpref: 100
                Router ID: 192.0.2.6
         BGP    Preference: 170/-101
                Route Distinguisher: 64497:1
                Next hop type: Indirect
                Address: 0x92413a8
                Next-hop reference count: 6
                Source: 192.0.2.3
                Next hop type: Router, Next hop index: 1322
                Next hop: 10.1.1.17 via ge-2/0/1.0, selected
                Label operation: Push 299840, Push 299776(top)
                Label TTL action: prop-ttl, prop-ttl(top)
                Session Id: 0x200005
                Protocol next hop: 192.0.2.3
                Push 299840
                Indirect next hop: 94100ec 1048584 INH Session ID: 0x20000b
                State: <Secondary NotBest Int Ext ProtectionCand>
                Inactive reason: Not Best in its group - Interior > Exterior > Exterior via
Interior
                Local AS: 64497 Peer AS: 64497
```

```
        Age: 10:31      Metric2: 1
        Validation State: unverified
        Task: BGP_64497.192.0.2.3+179
        Local AS: 64497 Peer AS: 64497
        Age: 10:31      Metric2: 1
        Validation State: unverified
        Task: BGP_64497.192.0.2.3+179
        AS path: 64498 I
        Communities: target:64497:1
        Import Accepted
        VPN Label: 299840
        Localpref: 100
        Router ID: 192.0.2.3
        Primary Routing Table bgp.l3vpn.0
        Indirect next hops: 1
                Protocol next hop: 192.0.2.3 Metric: 1
                Push 299840
                Indirect next hop: 94100ec 1048584 INH Session ID: 0x20000b
                Indirect path forwarding next hops: 1
                        Next hop type: Router
                        Next hop: 10.1.1.17 via ge-2/0/1.0
                        Session Id: 0x200005
                192.0.2.3/24 Originating RIB: inet.3
                  Metric: 1                       Node path count: 1
                  Forwarding nexthops: 1
                        Nexthop: 10.1.1.17 via ge-2/0/1.0
#Multipath Preference: 255
        Next hop type: List, Next hop index: 1048585
        Address: 0x944c154
        Next-hop reference count: 2
        Next hop: ELNH Address 0x9240a74 weight 0x1, selected
        equal-external-internal-type external
            Next hop type: Router, Next hop index: 994
            Address: 0x9240a74
            Next-hop reference count: 5
            Next hop: 10.1.1.26 via ge-2/0/2.0
        Next hop: ELNH Address 0x92413a8 weight 0x4000
        equal-external-internal-type internal
            Next hop type: Indirect
            Address: 0x92413a8
            Next-hop reference count: 6
            Protocol next hop: 192.0.2.3
            Push 299840
```

```
                      Indirect next hop: 94100ec 1048584 INH Session ID: 0x20000b
                          Next hop type: Router, Next hop index: 1322
                          Address: 0x9241310
                          Next-hop reference count: 4
                          Next hop: 10.1.1.17 via ge-2/0/1.0
                          Label operation: Push 299840, Push 299776(top)
                          Label TTL action: prop-ttl, prop-ttl(top)
                  State: <ForwardingOnly Int Ext>
                  Inactive reason: Forwarding use only
                  Age: 10:31
                  Validation State: unverified
                  Task: RT
                  Announcement bits (1): 0-KRT
                  AS path: 64498 I
```

The output shows that the weight (**0x1**) assigned to the PE3-CE2 path is preferable over the (**0x4000**) weight that is assigned to the PE2-CE2 path. A lower weight value takes precedence over a higher weight value. This confirms that the PE3-CE2 path is protected by the PE2-CE2 path'

Meaning

The provider edge link between Routers PE3 and CE2 is protected.

Example: Configuring Provider Edge Link Protection for BGP Labeled Unicast Paths

IN THIS SECTION

This example shows how to configure a labeled unicast protection path that can be used in case of a link failure in a carrier-of-carriers topology.

Requirements

This example uses the following hardware and software components:

- M Series Multiservice Edge Routers, MX Series 5G Universal Routing Platforms, or T Series Core Routers
- Junos OS Release 13.3 or later

Overview

IN THIS SECTION

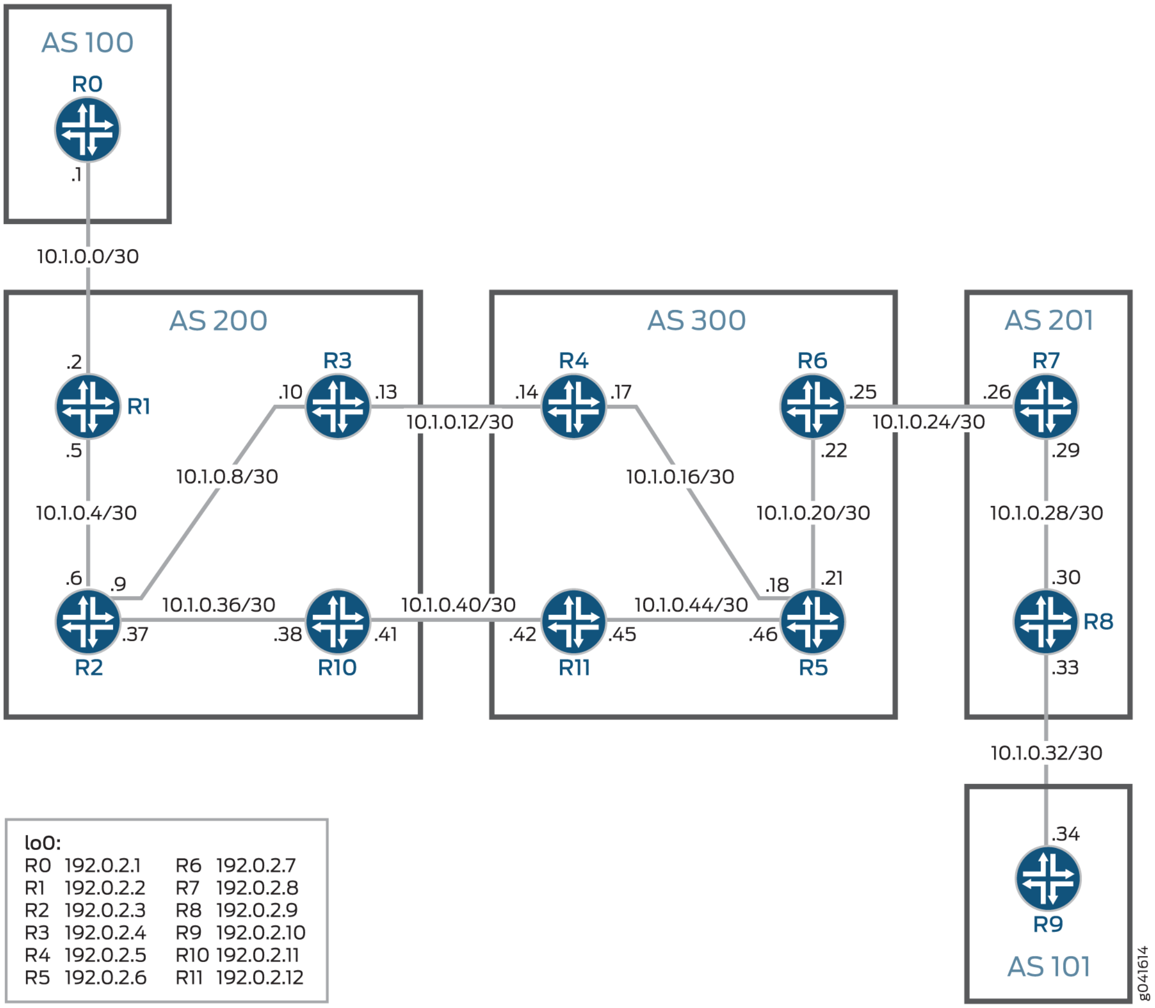
This example shows how to configure labeled-unicast link protection in a Layer 3 VPN.

Topology

In this example, a carrier-of-carriers topology is set up by configuring two customer edge devices and eight service provider edge devices in five autonomous systems. The CE devices are configured in AS100 and AS101. The PE devices are configured in AS200, AS300, and AS201.

Figure 84 on page 1092 shows the topology used in this example.

Figure 84: Labeled Unicast Link Protection in a Layer 3 VPN



The aim of this example is to protect the provider edge link between Routers R4 and R3. Protection is configured on the primary link between R4 and R3 such that the traffic can be routed through the backup link (R11 to R10) when the primary link goes down.

NOTE: Protection can also be configured on the secondary link between R11 and R10 so that if that link becomes the primary link and the R4-R3 link becomes secondary, the R11-R10 link will be protected as well.

Configuration

IN THIS SECTION

CLI Quick Configuration

To quickly configure this example, copy the following commands, paste them into a text file, remove any line breaks, change any details necessary to match your network configuration, and then copy and paste the commands into the CLI at the `[edit]` hierarchy level.

NOTE: Protection is added to the configuration only after the initial configuration is committed and BGP has installed the best path in the forwarding table.

Router R0

```
set interfaces ge-2/0/0 unit 0 description toR1
set interfaces ge-2/0/0 unit 0 family inet address 10.1.0.1/30
set interfaces ge-2/0/0 unit 0 family iso
set interfaces ge-2/0/0 unit 0 family mpls
set interfaces lo0 unit 0 family inet address 192.0.2.1/24
set interfaces lo0 unit 0 family iso address 47.0005.80ff.f800.0000.0108.0001.0102.5507.2056.00
set routing-options router-id 192.0.2.1
set routing-options autonomous-system 100
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set protocols ospf area 0.0.0.0 interface ge-2/0/0.0 metric 10
```

Router R1

```
set interfaces ge-2/0/0 unit 0 description toR0
set interfaces ge-2/0/0 unit 0 family inet address 10.1.0.2/30
set interfaces ge-2/0/0 unit 0 family iso
set interfaces ge-2/0/0 unit 0 family mpls
set interfaces ge-2/0/1 unit 0 description toR2
```

```
set interfaces ge-2/0/1 unit 0 family inet address 10.1.0.5/30
set interfaces ge-2/0/1 unit 0 family iso
set interfaces ge-2/0/1 unit 0 family mpls
set interfaces lo0 unit 0 family inet address 192.0.2.2/24
set interfaces lo0 unit 0 family iso address 47.0005.80ff.f800.0000.0108.0001.0102.5507.2052.00
set routing-options router-id 192.0.2.2
set routing-options autonomous-system 200
set protocols mpls interface all
set protocols ldp track-igp-metric
set protocols ldp interface ge-2/0/1.0
set protocols ldp interface lo0.0
set protocols ospf traffic-engineering
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set protocols ospf area 0.0.0.0 interface ge-2/0/1.0 metric 10
set protocols bgp group toR8 local-address 192.0.2.2
set protocols bgp group toR8 type external
set protocols bgp group toR8 multihop ttl 10
set protocols bgp group toR8 family inet-vpn unicast
set protocols bgp group toR8 neighbor 192.0.2.9 peer-as 201
set policy-options policy-statement child_vpn_routes from protocol bgp
set policy-options policy-statement child_vpn_routes then accept
set routing-instances customer-provider-vpn instance-type vrf
set routing-instances customer-provider-vpn interface ge-2/0/0.0
set routing-instances customer-provider-vpn route-distinguisher 192.0.2.4:1
set routing-instances customer-provider-vpn vrf-target target:200:1
set routing-instances customer-provider-vpn protocols ospf export child_vpn_routes
set routing-instances customer-provider-vpn protocols ospf area 0.0.0.0 interface ge-2/0/0.0
```

Router R2

```

set interfaces ge-2/0/0 unit 0 description toR1
set interfaces ge-2/0/0 unit 0 family inet address 10.1.0.6/30
set interfaces ge-2/0/0 unit 0 family iso
set interfaces ge-2/0/0 unit 0 family mpls
set interfaces ge-2/0/1 unit 0 description toR3
set interfaces ge-2/0/1 unit 0 family inet address 10.1.0.9/30
set interfaces ge-2/0/1 unit 0 family iso
set interfaces ge-2/0/1 unit 0 family mpls
set interfaces ge-2/0/2 unit 0 description toR10
set interfaces ge-2/0/2 unit 0 family inet address 10.1.0.37/30
set interfaces ge-2/0/2 unit 0 family iso
set interfaces ge-2/0/2 unit 0 family mpls
```

```
set interfaces lo0 unit 0 family inet address 192.0.2.3/24
set interfaces lo0 unit 0 family iso address 47.0005.80ff.f800.0000.0108.0001.0102.5507.2046.00
set routing-options router-id 192.0.2.3
set routing-options autonomous-system 200
set protocols mpls interface all
set protocols ldp track-igp-metric
set protocols ldp interface ge-2/0/0.0
set protocols ldp interface ge-2/0/1.0
set protocols ldp interface ge-2/0/2.0
set protocols ldp interface lo0.0
set protocols ospf traffic-engineering
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set protocols ospf area 0.0.0.0 interface ge-2/0/0.0 metric 10
set protocols ospf area 0.0.0.0 interface ge-2/0/1.0 metric 10
set protocols ospf area 0.0.0.0 interface ge-2/0/2.0 metric 10
```

Router R3

```
set interfaces ge-2/0/0 unit 0 description toR2
set interfaces ge-2/0/0 unit 0 family inet address 10.1.0.10/30
set interfaces ge-2/0/0 unit 0 family iso
set interfaces ge-2/0/0 unit 0 family mpls
set interfaces ge-2/0/1 unit 0 description toR4
set interfaces ge-2/0/1 unit 0 family inet address 10.1.0.13/30
set interfaces ge-2/0/1 unit 0 family iso
set interfaces ge-2/0/1 unit 0 family mpls
set interfaces lo0 unit 0 family inet address 192.0.2.4/24
set interfaces lo0 unit 0 family iso address 47.0005.80ff.f800.0000.0108.0001.0102.5507.2045.00
set routing-options router-id 192.0.2.4
set routing-options autonomous-system 200
set protocols mpls traffic-engineering bgp-igp
set protocols mpls interface all
set protocols ldp track-igp-metric
set protocols ldp egress-policy from-bgp
set protocols ldp interface ge-2/0/0.0
set protocols ldp interface lo0.0
set protocols ospf traffic-engineering
set protocols ospf export from-bgp
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set protocols ospf area 0.0.0.0 interface ge-2/0/0.0 metric 10
set protocols bgp group toR4 type external
set protocols bgp group toR4 import send-local
```

```
set protocols bgp group toR4 family inet labeled-unicast
set protocols bgp group toR4 export send-local
set protocols bgp group toR4 neighbor 10.1.0.14 peer-as 300
set policy-options policy-statement from-bgp from protocol bgp
set policy-options policy-statement from-bgp then metric add 100
set policy-options policy-statement from-bgp then accept
set policy-options policy-statement send-local term 2 from metric 100
set policy-options policy-statement send-local term 2 then reject
set policy-options policy-statement send-local then accept
```

Router R4

```
set interfaces ge-2/0/0 unit 0 description toR3
set interfaces ge-2/0/0 unit 0 family inet address 10.1.0.14/30
set interfaces ge-2/0/0 unit 0 family iso
set interfaces ge-2/0/0 unit 0 family mpls
set interfaces ge-2/0/1 unit 0 description toR5
set interfaces ge-2/0/1 unit 0 family inet address 10.1.0.17/30
set interfaces ge-2/0/1 unit 0 family iso
set interfaces ge-2/0/1 unit 0 family mpls
set interfaces lo0 unit 0 family inet address 192.0.2.5/24
set interfaces lo0 unit 0 family iso address 47.0005.80ff.f800.0000.0108.0001.0102.5507.2049.00
set policy-options policy-statement lb then load-balance per-packet
set routing-options router-id 192.0.2.5
set routing-options autonomous-system 300
set routing-options forwarding-table export lb
set protocols mpls interface all
set protocols ldp track-igp-metric
set protocols ldp interface ge-2/0/1.0
set protocols ldp interface lo0.0
set protocols isis level 1 disable
set protocols isis level 2 wide-metrics-only
set protocols isis interface ge-2/0/1.0 level 2 metric 10
set protocols isis interface lo0.0 passive
set protocols bgp group parent-vpn-peers type internal
set protocols bgp group parent-vpn-peers local-address 192.0.2.5
set protocols bgp group parent-vpn-peers family inet-vpn unicast
set protocols bgp group parent-vpn-peers neighbor 192.0.2.7
set protocols bgp group parent-vpn-peers neighbor 192.0.2.12
set routing-instances coc-provider-vpn instance-type vrf
set routing-instances coc-provider-vpn interface ge-2/0/0.0
set routing-instances coc-provider-vpn interface ge-2/0/2.0
```

```
set routing-instances coc-provider-vpn route-distinguisher 192.0.2.5:1
set routing-instances coc-provider-vpn vrf-target target:300:1
set routing-instances coc-provider-vpn protocols bgp group toR3 type external
set routing-instances coc-provider-vpn protocols bgp group toR3 family inet labeled-unicast per-
prefix-label
set routing-instances coc-provider-vpn protocols bgp group toR3 neighbor 10.1.0.13 peer-as 200
```

Router R5

```
set interfaces ge-2/0/0 unit 0 description toR4
set interfaces ge-2/0/0 unit 0 family inet address 10.1.0.18/30
set interfaces ge-2/0/0 unit 0 family iso
set interfaces ge-2/0/0 unit 0 family mpls
set interfaces ge-2/0/1 unit 0 description toR6
set interfaces ge-2/0/1 unit 0 family inet address 10.1.0.21/30
set interfaces ge-2/0/1 unit 0 family iso
set interfaces ge-2/0/1 unit 0 family mpls
set interfaces ge-2/0/2 unit 0 description toR11
set interfaces ge-2/0/2 unit 0 family inet address 10.1.0.46/30
set interfaces ge-2/0/2 unit 0 family iso
set interfaces ge-2/0/2 unit 0 family mpls
set interfaces lo0 unit 0 family inet address 192.0.2.6/24
set interfaces lo0 unit 0 family iso address 47.0005.80ff.f800.0000.0108.0001.0102.5507.2050.00
set routing-options router-id 192.0.2.6
set routing-options autonomous-system 300
set protocols mpls interface all
set protocols ldp track-igp-metric
set protocols ldp interface ge-2/0/0.0
set protocols ldp interface ge-2/0/1.0
set protocols ldp interface ge-2/0/2.0
set protocols ldp interface lo0.0
set protocols isis level 1 disable
set protocols isis level 2 wide-metrics-only
set protocols isis interface ge-2/0/0.0 level 2 metric 10
set protocols isis interface ge-2/0/1.0 level 2 metric 10
set protocols isis interface ge-2/0/2.0 level 2 metric 10
set protocols isis interface lo0.0 passive
```

Router R6

```
set interfaces ge-2/0/0 unit 0 description toR5
set interfaces ge-2/0/0 unit 0 family inet address 10.1.0.22/30
set interfaces ge-2/0/0 unit 0 family iso
set interfaces ge-2/0/0 unit 0 family mpls
set interfaces ge-2/0/1 unit 0 description toR7
set interfaces ge-2/0/1 unit 0 family inet address 10.1.0.25/30
set interfaces ge-2/0/1 unit 0 family iso
set interfaces ge-2/0/1 unit 0 family mpls
set interfaces lo0 unit 0 family inet address 192.0.2.7/24
set interfaces lo0 unit 0 family iso address 47.0005.80ff.f800.0000.0108.0001.0102.5507.2048.00
set routing-options router-id 192.0.2.7
set routing-options autonomous-system 300
set protocols mpls interface all
set protocols ldp track-igp-metric
set protocols ldp interface ge-2/0/0.0
set protocols ldp interface lo0.0
set protocols isis level 1 disable
set protocols isis level 2 wide-metrics-only
set protocols isis interface ge-2/0/0.0 level 2 metric 10
set protocols isis interface lo0.0 passive
set protocols bgp group parent-vpn-peers type internal
set protocols bgp group parent-vpn-peers local-address 192.0.2.7
set protocols bgp group parent-vpn-peers family inet-vpn unicast
set protocols bgp group parent-vpn-peers neighbor 192.0.2.5
set protocols bgp group parent-vpn-peers neighbor 192.0.2.12
set routing-instances coc-provider-vpn instance-type vrf
set routing-instances coc-provider-vpn interface ge-2/0/1.0
set routing-instances coc-provider-vpn route-distinguisher 192.0.2.7:1
set routing-instances coc-provider-vpn vrf-target target:300:1
set routing-instances coc-provider-vpn protocols bgp group toR7 family inet labeled-unicast per-prefix-label
set routing-instances coc-provider-vpn protocols bgp group toR7 type external
set routing-instances coc-provider-vpn protocols bgp group toR7 neighbor 10.1.0.26 peer-as 201
```

Router R7

```
set interfaces ge-2/0/0 unit 0 description toR6
set interfaces ge-2/0/0 unit 0 family inet address 10.1.0.26/30
set interfaces ge-2/0/0 unit 0 family iso
```

```
set interfaces ge-2/0/0 unit 0 family mpls
set interfaces ge-2/0/1 unit 0 description toR8
set interfaces ge-2/0/1 unit 0 family inet address 10.1.0.29/30
set interfaces ge-2/0/1 unit 0 family iso
set interfaces ge-2/0/1 unit 0 family mpls
set interfaces lo0 unit 0 family inet address 192.0.2.8/24
set interfaces lo0 unit 0 family iso address 47.0005.80ff.f800.0000.0108.0001.0102.5507.2054.00
set routing-options router-id 192.0.2.8
set routing-options autonomous-system 201
set protocols mpls traffic-engineering bgp-igp
set protocols mpls interface all
set protocols ldp track-igp-metric
set protocols ldp egress-policy from-bgp
set protocols ldp interface ge-2/0/1.0
set protocols ldp interface lo0.0
set protocols ospf traffic-engineering
set protocols ospf export from-bgp
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set protocols ospf area 0.0.0.0 interface ge-2/0/1.0 metric 10
set protocols bgp group toR6 type external
set protocols bgp group toR6 import send-all
set protocols bgp group toR6 family inet labeled-unicast
set protocols bgp group toR6 export send-all
set protocols bgp group toR6 neighbor 10.1.0.25 peer-as 300
set policy-options policy-statement from-bgp from protocol bgp
set policy-options policy-statement from-bgp then accept
set policy-options policy-statement send-all then accept
```

Router R8

```
set interfaces ge-2/0/0 unit 0 description toR7
set interfaces ge-2/0/0 unit 0 family inet address 10.1.0.30/30
set interfaces ge-2/0/0 unit 0 family iso
set interfaces ge-2/0/0 unit 0 family mpls
set interfaces ge-2/0/1 unit 0 description toR9
set interfaces ge-2/0/1 unit 0 family inet address 10.1.0.33/30
set interfaces ge-2/0/1 unit 0 family iso
set interfaces ge-2/0/1 unit 0 family mpls
set interfaces lo0 unit 0 family inet address 192.0.2.9/24
set interfaces lo0 unit 0 family iso address 47.0005.80ff.f800.0000.0108.0001.0102.5507.2053.00
set routing-options router-id 192.0.2.9
set routing-options autonomous-system 201
```

```
set protocols mpls interface all
set protocols ldp track-igp-metric
set protocols ldp interface ge-2/0/0.0
set protocols ldp interface lo0.0
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set protocols ospf area 0.0.0.0 interface ge-2/0/0.0 metric 10
set protocols bgp group toR1 local-address 192.0.2.9
set protocols bgp group toR1 type external
set protocols bgp group toR1 multihop ttl 10
set protocols bgp group toR1 family inet-vpn unicast
set protocols bgp group toR1 neighbor 192.0.2.2 peer-as 200
set policy-options policy-statement child_vpn_routes from protocol bgp
set policy-options policy-statement child_vpn_routes then accept
set routing-instances customer-provider-vpn instance-type vrf
set routing-instances customer-provider-vpn interface ge-2/0/1.0
set routing-instances customer-provider-vpn route-distinguisher 192.0.2.9:1
set routing-instances customer-provider-vpn vrf-target target:200:1
set routing-instances customer-provider-vpn protocols ospf export child_vpn_routes
set routing-instances customer-provider-vpn protocols ospf area 0.0.0.0 interface ge-2/0/1.0
```

Router R9

```
set interfaces ge-2/0/0 unit 0 description toR8
set interfaces ge-2/0/0 unit 0 family inet address 10.1.0.34/30
set interfaces ge-2/0/0 unit 0 family iso
set interfaces ge-2/0/0 unit 0 family mpls
set interfaces lo0 unit 0 family inet address 192.0.2.10/24
set interfaces lo0 unit 0 family iso address 47.0005.80ff.f800.0000.0108.0001.0102.5507.2047.00
set routing-options router-id 192.0.2.10
set routing-options autonomous-system 101
set routing-options static route 198.51.100.1/24 discard
set protocols ospf export statics
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set protocols ospf area 0.0.0.0 interface ge-2/0/0.0 metric 10
set policy-options policy-statement statics from route-filter 198.51.100.1/24 exact
set policy-options policy-statement statics then accept
```

Router R10

```
set interfaces ge-2/0/0 unit 0 description toR2
set interfaces ge-2/0/0 unit 0 family inet address 10.1.0.38/30
```

```
set interfaces ge-2/0/0 unit 0 family iso
set interfaces ge-2/0/0 unit 0 family mpls
set interfaces ge-2/0/1 unit 0 description toR11
set interfaces ge-2/0/1 unit 0 family inet address 10.1.0.41/30
set interfaces ge-2/0/1 unit 0 family iso
set interfaces ge-2/0/1 unit 0 family mpls
set interfaces lo0 unit 0 family inet address 192.0.2.11.24
set interfaces lo0 unit 0 family iso address 47.0005.80ff.f800.0000.0108.0001.0102.5507.2061.00
set routing-options router-id 192.0.2.11
set routing-options autonomous-system 200
set protocols mpls traffic-engineering bgp-igp
set protocols mpls interface all
set protocols ldp track-igp-metric
set protocols ldp egress-policy from-bgp
set protocols ldp interface ge-2/0/0.0
set protocols ldp interface lo0.0
set protocols ospf traffic-engineering
set protocols ospf export from-bgp
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set protocols ospf area 0.0.0.0 interface ge-2/0/0.0 metric 10
set protocols bgp group toR4 type external
set protocols bgp group toR4 import send-local
set protocols bgp group toR4 family inet labeled-unicast
set protocols bgp group toR4 export send-local
set protocols bgp group toR4 neighbor 10.1.0.42 peer-as 300
set protocols bgp group toR4 inactive: neighbor 10.1.0.50 peer-as 300
set policy-options policy-statement from-bgp from protocol bgp
set policy-options policy-statement from-bgp then metric add 100
set policy-options policy-statement from-bgp then accept
set policy-options policy-statement send-local term 2 from metric 100
set policy-options policy-statement send-local term 2 then reject
set policy-options policy-statement send-local then accept
```

Router R11

```
set interfaces ge-2/0/0 unit 0 description toR10
set interfaces ge-2/0/0 unit 0 family inet address 10.1.0.42/30
set interfaces ge-2/0/0 unit 0 family iso
set interfaces ge-2/0/0 unit 0 family mpls
set interfaces ge-2/0/1 unit 0 description toR5
set interfaces ge-2/0/1 unit 0 family inet address 10.1.0.45/30
set interfaces ge-2/0/1 unit 0 family iso
```

```
set interfaces ge-2/0/1 unit 0 family mpls
set interfaces lo0 unit 0 family inet address 192.0.2.12/24
set interfaces lo0 unit 0 family iso address 47.0005.80ff.f800.0000.0108.0001.0102.5507.2062.00
set routing-options router-id 192.0.2.12
set routing-options autonomous-system 300
set protocols mpls interface all
set protocols ldp track-igp-metric
set protocols ldp interface ge-2/0/1.0
set protocols ldp interface lo0.0
set protocols isis level 1 disable
set protocols isis level 2 wide-metrics-only
set protocols isis interface ge-2/0/1.0 level 2 metric 10
set protocols isis interface lo0.0 passive
set protocols bgp group parent-vpn-peers type internal
set protocols bgp group parent-vpn-peers local-address 192.0.2.12
set protocols bgp group parent-vpn-peers family inet-vpn unicast
set protocols bgp group parent-vpn-peers neighbor 192.0.2.7
set protocols bgp group parent-vpn-peers neighbor 192.0.2.12
set routing-instances coc-provider-vpn instance-type vrf
set routing-instances coc-provider-vpn interface ge-2/0/0.0
set routing-instances coc-provider-vpn route-distinguisher 192.0.2.12:1
set routing-instances coc-provider-vpn vrf-target target:300:1
set routing-instances coc-provider-vpn protocols bgp group toR10 family inet labeled-unicast per-
prefix-label
set routing-instances coc-provider-vpn protocols bgp group toR10 type external
set routing-instances coc-provider-vpn protocols bgp group toR10 neighbor 10.1.0.41 peer-as 200
```

Configuring Provider Edge Link Protection in Layer 3 VPNs

Step-by-Step Procedure

The following example requires that you navigate various levels in the configuration hierarchy. For information about navigating the CLI, see *Using the CLI Editor in Configuration Mode* in the CLI User Guide.

To configure labeled unicast link protection:

1. Configure the router interfaces.

   ```
   [edit interfaces]
   user@R4# set ge-2/0/0 unit 0 description toR3
   user@R4# set ge-2/0/0 unit 0 family inet address 10.1.0.14/30
   ```

```
user@R4# set ge-2/0/0 unit 0 family iso
user@R4# set ge-2/0/0 unit 0 family mpls
```

```
user@R4# set ge-2/0/1 unit 0 description toR5
user@R4# set ge-2/0/1 unit 0 family inet address 10.1.0.17/30
user@R4# set ge-2/0/1 unit 0 family iso
user@R4# set ge-2/0/1 unit 0 family mpls
```

```
user@R4# set lo0 unit 0 family inet address 192.0.2.5/24
user@R4# set lo0 unit 0 family iso address 47.0005.80ff.f800.0000.0108.0001.0102.5507.2049.00
```

Similarly, configure the interfaces on all other routers.

2. Configure the routing policy options on R4.

```
[edit policy-options]
user@R4# set policy-statement lb then load-balance per-packet
```

Similarly, configure the policy options on routers R1, R3, R7, R8, R9, and R10 for this example.

3. Configure the router ID, autonomous system (AS) number, and any other routing options.

```
[edit routing-options]
user@R4# set router-id 192.0.2.5
user@R4# set autonomous-system 300
user@R4# set forwarding-table export lb
```

Similarly, configure the router ID, AS number, and any other routing options for all other routers. In this example, the router ID is chosen to be identical to the loopback address configured on the router.

4. Configure MPLS and LDP on Router R4.

```
[edit protocols]
user@R4# set mpls interface all
user@R4# set ldp track-igp-metric
user@R4# set ldp interface ge-2/0/1.0
user@R4# set ldp interface lo0.0
```

Similarly, configure MPLS and LDP on all other routers except R0 and R9.

5. Configure an IGP on the core-facing interfaces of Router R4.

```
[edit protocols isis]
user@R4# set level 1 disable
user@R4# set level 2 wide-metrics-only
user@R4# set interface ge-2/0/1.0 level 2 metric 10
user@R4# set interface lo0.0 passive
```

Similarly, configure other routers (IS-IS on R5, R6, and R11 and OSPF on all other routers in this example).

6. Configure BGP on Router R4.

```
[edit protocols bgp group parent-vpn-peers]
user@R4# set type internal
user@R4# set local-address 192.0.2.5
user@R4# set family inet-vpn unicast
user@R4# set neighbor 192.0.2.7
user@R4# set neighbor 192.0.2.12
```

Similarly, configure BGP on routers R1, R3, R6, R7, R8, R10, and R11.

7. Configure a VPN routing and forwarding (VRF) instance on Router R4 to create a Layer 3 VPN.

```
[edit routing-instances coc-provider-vpn]
user@R4# set instance-type vrf
user@R4# set interface ge-2/0/0.0
user@R4# set interface ge-2/0/2.0
user@R4# set route-distinguisher 192.0.2.5:1
user@R4# set vrf-target target:300:1
```

```
[edit routing-instances coc-provider-vpn protocols bgp group toR3]
user@R4# set type external
user@R4# set family inet labeled-unicast per-prefix-label
user@R4# set neighbor 10.1.0.13 peer-as 200
```

Similarly, configure other VRF routing instances on R1, R6, R8, and R11.

Results

From configuration mode, confirm your configuration by entering the `show interfaces`, `show policy-options`, `show routing-options`, `show protocols`, and `show routing-instances` commands.

If the output does not display the intended configuration, repeat the instructions in this example to correct the configuration.

```
user@R4# show interfaces
ge-2/0/0 {
    unit 0 {
        description toR3;
        family inet {
            address 10.1.0.14/30;
        }
        family iso;
        family mpls;
    }
}
ge-2/0/1 {
    unit 0 {
        description toR5;
        family inet {
            address 10.1.0.17/30;
        }
        family iso;
        family mpls;
    }
}

lo0 {
    unit 0 {
        family inet {
            address 192.0.2.5/24;
        }
        family iso {
            address 47.0005.80ff.f800.0000.0108.0001.0102.5507.2049.00;
        }
```

```
    }
}
```

```
user@R4# show policy-options
policy-statement lb {
    then {
        load-balance per-packet;
    }
}
```

```
user@R4# show routing-options
router-id 192.0.2.5;
autonomous-system 300;
forwarding-table {
    export  lb;
}
```

```
user@R4# show protocols
mpls {
    interface all;
}
ldp {
    track-igp-metric;
    interface ge-2/0/1.0;
    interface lo0.0;
}
isis {
    level 1 disable;
    level 2 wide-metrics-only;
    interface ge-2/0/1.0 {
        level 2 metric 10;
    }
    interface lo0.0 {
        passive;
    }
}
bgp {
    group parent-vpn-peers {
        type internal;
```

```
        local-address 192.0.2.5;
        family inet-vpn {
            unicast;
        }
        neighbor 192.0.2.7;
        neighbor 192.0.2.12;
    }
}
```

```
user@R4# show routing-instances
coc-provider-vpn {
    instance-type vrf;
    interface ge-2/0/0.0;
    interface ge-2/0/2.0;
    route-distinguisher 192.0.2.5:1;
    vrf-target target:300:1;
    protocols {
        bgp {
            group toR3 {
                type external;
                family inet {
                    labeled-unicast {
                        per-prefix-label;
                    }
                }
                neighbor 10.1.0.13 {
                    peer-as 200;
                }
            }
        }
    }
}
```

If you are done configuring the router, enter `commit` from configuration mode.

Repeat the procedure for every router in this example, using the appropriate interface names and addresses for each router.

Verification

IN THIS SECTION

Confirm that the configuration is working properly.

Enabling Protection

Purpose

Enable protection on R4 to request protection for the link from R4 to R3.

Action

1. Add the `protection` statement at the `[edit routing-instances` *`instance-name`* `protocols bgp group` *`group-name`* `family inet labeled-unicast]` hierarchy level.

   ```
   [edit routing-instances coc-provider-vpn protocols bgp group toR3]
   user@R4# set  family inet labeled-unicast protection
   ```

2. Verify and commit the configuration.

   ```
   type external;
   family inet {
       labeled-unicast {
           per-prefix-label;
           protection;
       }
   }
   neighbor 10.1.0.13 {
       peer-as 200;
   }
   ```

Verifying Multipath Entries

Purpose

Verify that R4 has a multipath entry with two entries.

Action

From operational mode on Router R4, run the `show route 192.0.2.2` command to check the route to R1.

```
user@R4> show route 192.0.2.2
#[Multipath/255] 00:02:44, metric 20
        > to 10.1.0.13 via ge-2/0/0.0, Push 408592
          to 10.1.0.18 via ge-2/0/1.0, Push 299856, Push 299792(top)
```

Verifying That Multipath Entries Have Different Weights

Purpose

Verify that the two routes in the multipath entry have different weights, with the first entry having a weight of 0x1 and the second having a weight of 0x4000.

Action

From operational mode on Router R4, run the `show route 192.0.2.2 detail` command to check the route to R1.

```
user@R4> show route 192.0.2.2 detail
        #Multipath Preference: 255
                Next hop type: List, Next hop index: 1048609
                Address: 0x92f058c
                Next-hop reference count: 4
                Next hop: ELNH Address 0x92c48ac weight 0x1, selected
                equal-external-internal-type external
                    Next hop type: Router, Next hop index: 1603
                    Address: 0x92c48ac
                    Next-hop reference count: 2
                    Next hop: 10.1.0.13 via ge-2/0/0.0
                    Label operation: Push 408592
                    Label TTL action: prop-ttl
```

```
                Next hop: ELNH Address 0x92c548c weight 0x4000
                equal-external-internal-type internal
                    Next hop type: Indirect
                    Address: 0x92c548c
                    Next-hop reference count: 3
                    Protocol next hop: 192.0.2.12
                    Push 299856
                    Indirect next hop: 0x9380f40 1048608 INH Session ID: 0x10001a
                        Next hop type: Router, Next hop index: 1586
                        Address: 0x92c5440
                        Next-hop reference count: 3
                        Next hop: 10.1.0.18 via ge-2/0/1.0
                        Label operation: Push 299856, Push 299792(top)
                        Label TTL action: prop-ttl, prop-ttl(top)
                State: <ForwardingOnly Int Ext>
                Inactive reason: Forwarding use only
                Age: 3:38       Metric: 20
                Validation State: unverified
                Task: RT
                Announcement bits (1): 0-KRT
                AS path: 200 I
```

Understanding Host Fast Reroute

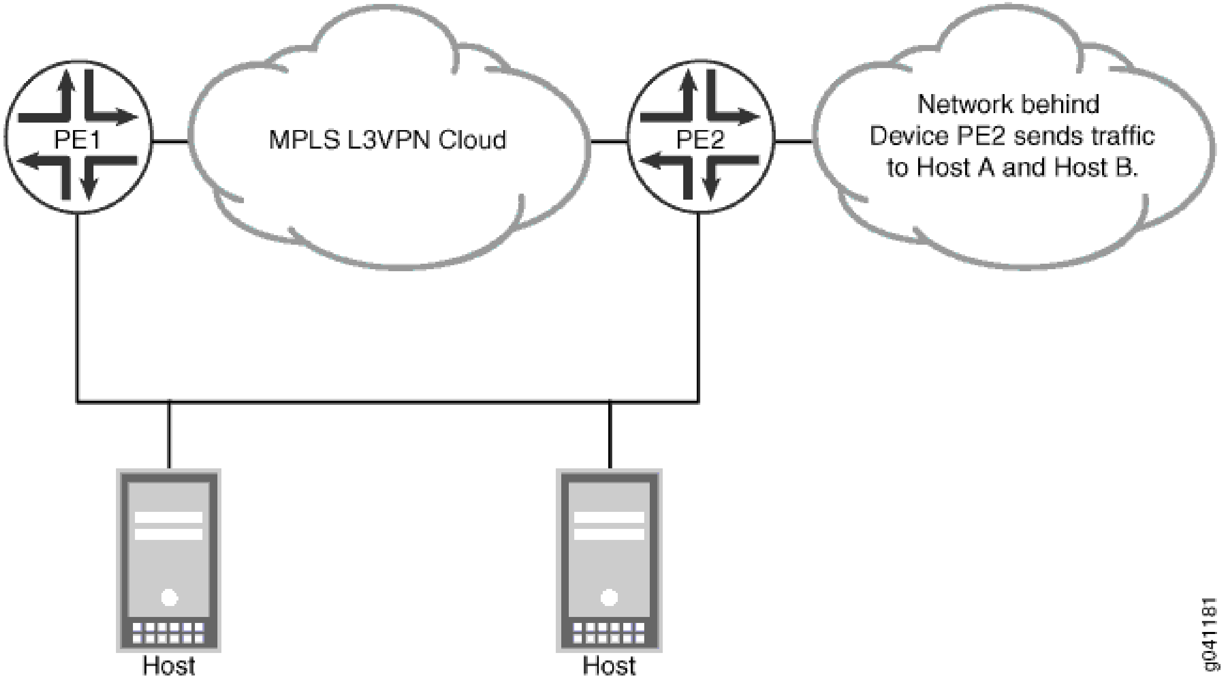
IN THIS SECTION

Host fast reroute (HFRR) adds a precomputed protection path into the Packet Forwarding Engine (PFE), such that if a link between a provider edge device and a server farm becomes unusable for forwarding, the PFE can use another path without having to wait for the router or the protocols to provide updated forwarding information. This precomputed protection path is often called a repair or a backup path.

HFRR is a technology that protects IP endpoints on multipoint interfaces, such as Ethernet. This technology is important in datacenters where fast service restoration for server endpoints is critical. After an interface or a link goes down, HFRR enables the local repair time to be approximately 50 milliseconds.

Consider the network topology shown in Figure 85 on page 1111.

Figure 85: Host Fast Reroute



Routing devices create host route forwarding entries triggered by the Address Resolution Protocol (ARP) and IPv6 Neighbor Discovery Protocol (NDP). HFRR augments the host routes with backup next hops supplied by routing protocols. These backup next hops enable arriving traffic to keep flowing while the network reconverges.

Traffic flows from networks connected to the provider edge devices, PE1 and PE2, to host A and host B. This traffic is protected with HFRR. If the link goes down between device PE2 and the host servers, traffic is rerouted through device PE1 to the host servers. In the topology, host A and host B represent LAN PCs, collectively known as a server farm. The PE devices are routers with a Layer 3 VPN configured between them. Device PE1 learns about the directly connected hosts by way of ARP or the IPv6 NDP.

Device PE2 also has information about the server farm network and advertises this information to Device PE1. This advertisement is transmitted through the Layer 3 VPN using internal BGP (IBGP). On Devices PE1 and PE2, this route is considered a direct route to the server farm subnet.

Device PE1 uses the host routes learned through ARP and NDP to send traffic to the host machines in the server farm. If the link between Device PE1 and the server farm is disrupted and if HFRR is not configured, the routing device finds the next best route, which is the IBGP route. This implementation results in traffic loss for an interval until the update occurs and the network reconverges. HFRR configured on Device PE1 resolves this issue by augmenting the ARP and NDP routes with a backup path so that traffic can continue to be forwarded without interruption.

The backup path in this particular topology is the IBGP Layer 3 VPN route. In an actual deployment, Device PE2 can also configure link protection for its directly connected server farm network, and Device PE1 can advertise reachability to the server farm through itself using the Layer 3 VPN routes to Device PE2. Therefore, HFRR should be enabled on both Device PE1 and Device PE2. Also, Device PE1 and Device PE2 should both advertise reachability to the server farm through BGP.

A temporary routing loop can develop between the PE devices if, for example, the link between Device PE1 to the server farm and the link between Device PE2 to the server farm both go down at same time. The loop can continue until BGP on both ends learns that the server farm subnet is down and withdraws the BGP routes.

ARP Prefix Limit and Blackout Supplementary Timeout

When you configure HFRR profiles, an optional ARP prefix limit sets a maximum for the number of ARP routes and, therefore, FRR routes created for each HFRR profile in the routing table. This limit prevents ARP attacks from exhausting the virtual memory on the routing devices. The ARP prefix limit does not limit ARP routes in the forwarding table. It does, however, limit the number of ARP routes that Junos OS reads for a profile and therefore limits the number of HFRR routes that the routing process (rpd) creates in the routing table and the forwarding table.

The ARP prefix limit is applied to each HFRR profile. It does not limit the total count of all ARP/HFRR routes in the routing table. It only limits the number of ARP/HFRR routes for each HFRR profile.

There are two configuration statements (`global-arp-prefix-limit` and `arp-prefix-limit`) that set the ARP prefix limit, one at the global `[edit routing-options host-fast-reroute]` hierarchy level and the other at the `[edit routing-instances` *`instance-name`* `routing-options interface` *`interface-name`*`]` hierarchy level, respectively. The global `global-arp-prefix-limit` statement sets a default ARP prefix limit for all HFRR profiles configured on the routing device. The `arp-prefix-limit` statement overrides the `global-arp-prefix-limit` for that HFRR profile for that protected interface.

When the number of ARP routes in an HFRR profile reaches 80% of the configured ARP prefix limit, a warning message is sent to the system log. The warning message is displayed for any subsequent ARP route added to that HFRR profile if the ARP prefixes remain at greater than 80% of the configured value.

When the number of ARP routes in an HFRR profile reaches 100% of the configured ARP prefix limit for an HFRR profile, another warning message is sent to the system log. When the number crosses the 100% threshold, the HFRR profile is deactivated. When this happens, all ARP/FRR routes are deleted from the routing table. FRR routes are deleted from forwarding table as well.

After the HFRR profile is deactivated, a blackout timer is started. The timeout value of this timer is the ARP cache timeout (kernel timeout) + the supplementary blackout timer.

There are global and per-HFRR CLI statements (`global-supplementary-blackout-timer` and `supplementary-blackout-timer`). The global value is at the `[edit routing-options host-fast-reroute]` hierarchy level and applies to all HFRR profiles on the routing device. The supplementary blackout timer configured for the routing-instance interface at the `[edit routing-instances` *`instance-name`* `routing-options interface` *`interface-name`*`]` hierarchy level overrides the global value for that HFRR profile only.

When the blackout timer expires, the HFRR profile is reactivated, and Junos OS relearns the ARP routes and re-creates the HFRR routes. If the ARP prefix limit is not exceeded again, the HFRR routes will be up.

If an HFRR profile is blocklisted and is in the deactivated state, a reevaluation of the ARP state is performed during every commit operation or whenever the routing process (rpd) is restarted with the `restart routing` command.

Primary Route and Backup Route Candidates

The primary route for the HFRR next hop is provided by the ARP and IPv6 NDP routes. These are /32 or /128 routes. The backup route is an exact prefix match of the address configured on the local interface. For example, if the local address configured is 10.0.0.5/24, the routing device looks for an exact match of prefix 10.0.0.0 with a prefix length of 24 for selection of backup route.

Constraints for backup route selection are as follows:

- Must be a prefix matching the same subnet address configured on the routing device’s HFRR-enabled interface.
- The remote end must not have route aggregation (also known as summarization) configured. For example, if the remote end combines two or more /24 subnets to advertise a subnet with a prefix length smaller than /24, the Junos OS does not select this summarized route to be a backup route.
- If there is another route in the routing table learned by another protocol with a longest-prefix match for the /32 or /128 (ARP or NDP) route, that route is not selected to be a backup candidate. For example, suppose that the local interface address is 10.0.0.5/24. Also suppose that the routing table contains an IBGP route with a prefix of 10.0.0.0/24 and an OSPF route with a prefix of 10.0.0.0/28. Even though the /28 route is a better route for certain prefixes in the subnet, the Junos OS does not consider 10.0.0.0/28 to be a backup candidate. The IBGP route becomes the backup candidate for all host routes. However, after the global repair, the OSPF route is used for forwarding.

In short, the backup candidate must be a route with the same prefix as the subnet local interface that you are protecting with HFRR.

Backup Path Selection Policy

Only Layer 3 VPN routes are considered for backup selection. HFRR uses the usual BGP path selection algorithm to select one best backup route. Only one backup path is selected. In case there are multiple backup path candidates, the selection algorithm selects the best backup path. HFRR provides only two paths, one primary and one backup at any point in time. If the selected backup path itself has two paths in it, then the first path in that backup next hop is used as the backup next hop for the HFRR route.

The primary path is installed with a weight of one. The backup path is installed with a weight of 0x4000. The backup path obviously must be a path through an interface that is not the same as the primary interface.

The backup route is looked up only in the routing table to which interface belongs. For IPv4, the Junos OS uses *routing-instance-name*.inet.0. For IPv6, the Junos OS looks in *routing-instance-name*.inet6.0.

Characteristics of HFRR Routes

The HFRR route is a forwarding-only route and is not used for route resolution. HFRR routes have host addresses, meaning that they have /32 or /128 as the prefix length. In the case of platforms with dual Routing Engines, the backup routing process (rpd) also creates HFRR routes. However, the backup outing process (rpd) does not install HFRR routes to a routing table until the backup becomes the primary after a Routing Engine switchover.

Also note that if an HFRR route is present in the routing table, the HFRR route is used for the unicast reverse-path-forwarding (uRPF) computation.

Removal of HFRR Routes

HFRR routes are deleted if the protected interface is deleted or deactivated in the configuration, if HFRR is configured on a routing instance and the routing instance is deactivated or deleted, or when the statement that enables HFRR (`link-protection (Host Fast Reroute)`) is deleted or deactivated. HFRR routes are deleted and readded when there is a catastrophic operation on routing the instance, such as when the routing process is restarted. HFRR routes are also be removed if all backup routes are deleted. such as when BGP withdraws routes or when BGP is deactivated or deleted.

After a protected interface goes down and if HFRR is deleted or deactivated, a timer starts with a timeout of 20 seconds. The HFRR route deletion occurs after the timer expires. This is to ensure that if the interface is flapping (quickly going up and down), the Junos OS does not unnecessarily perform route deletions and additions that cause traffic loss. This timer is used only when the interface is down or when the HFRR route is deleted or deactivated.

HFRR routes are purged immediately in the following cases:

- A backup route goes down and there are no other potential backup paths.

- An ARP delete message is received.
- The routing process (rpd) terminates.

Interfaces That Support HFRR

HFRR is allowed only on Ethernet interfaces. The commit operation fails if you configure HFRR on point-to-point interfaces.

Only interfaces configured under routing instance of type VPN routing and forwarding (VRF) are accepted. The commit operation fails if you configure HFRR on other types of routing instances.

When the following requirements are not met, the commit operation does not fail. However, the interface is not protected by HFRR, and the interface is marked inactive in the `show hfrr profiles` command output:

- HFRR is allowed only on numbered interfaces, meaning that an address must be assigned to the interface. You cannot, for example, configure IPv4on the interface with an address and IPv6 without an address.
- Interfaces that are configured for HFRR protection must be configured at the `[edit interfaces]` hierarchy level and also must be attached to the routing instance.
- The routing instance must have a virtual tunnel (VT) interface or the `vrf-table-label` statement included.

Another reason the interface might be marked inactive in the `show hfrr profiles` command output is when the interface is migrating from one instance to another, and the HFRR configuration is in the previous routing instance.

HFRR is not supported on overlapping logical units if they belong to the same routing instance, as shown here:

```
user@host # show interfaces
ge-0/0/2 {
    vlan-tagging;
    unit 0 {
        vlan-id 1;
        family inet {
            address 172.16.0.4/16;   # same subnet
        }
    }
    unit 1 {
        vlan-id 2;
        family inet {
```

```
            address 172.16.0.5/16;   # same subnet
        }
    }
}
```

If you configure overlapping subnets as shown here, and if you enable HFRR on both of the overlapping subnets, the routing protocol process (rpd) generates an RPD_ASSERT error.

SEE ALSO

Understanding BGP Path Selection

Example: Configuring Link Protection with Host Fast Reroute

IN THIS SECTION

This example shows you how to configure host fast reroute (HFRR). HFRR protects IP endpoints on multipoint interfaces, such as Ethernet.

Requirements

This example uses the following hardware and software components:

- Two provider edge (PE) devices and four provider (P) devices.
- The example assumes that hosts are present, behind the PE devices.
- The example assumes that at least one Layer 3 switch, such as an EX Series switch, is attached to the hosts.
- Junos OS 11.4R2 or later.

Overview

IN THIS SECTION

In this example, traffic flows to server hosts from networks connected to PE devices. This traffic is protected with HFRR. If the link goes down between one PE device and the server farm, traffic is rerouted to the server farm through the other PE device.

You can configure HFRR by adding the `link-protection` statement to the interface configuration in the routing instance.

```
[edit routing-instances cust1 routing-options]
set interface ge-4/1/0.0 link-protection (Host Fast Reroute)
```

We recommend that you include this statement on all PE devices that are connected to server farms through multipoint interfaces.

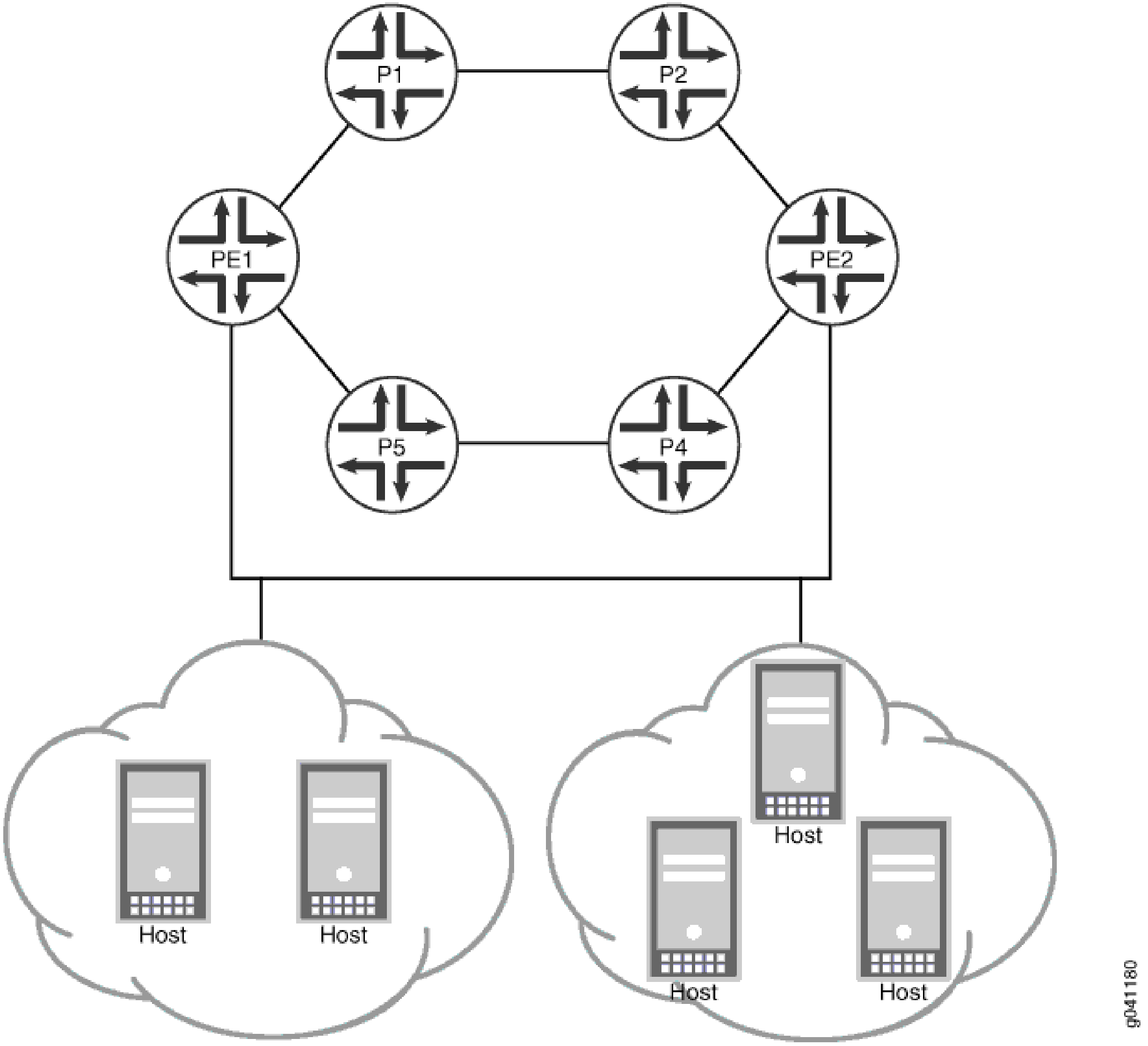
In this example, the PE devices advertise reachability to their server farms through Layer 3 VPN routes and BGP.

As optional settings, the PE devices have the high availability features Nonstop Active Routing and Virtual Router Redundancy Protocol (VRRP) configured. Nonstop active routing (NSR) enables a routing platform with redundant Routing Engines to switch from a primary Routing Engine to a backup Routing Engine without alerting peer nodes that a change has occurred and without losing routing and protocol information. VRRP provides for automatic assignment of available routers to participating hosts, thus increasing the availability and reliability of routing paths.

Topology Diagram

Figure 86 on page 1118 shows the topology used in this example.

Figure 86: Host Fast Reroute Topology



This example shows the configuration on all of the routing devices and shows the step-by-step procedure on Device PE1.

Configuration

IN THIS SECTION

CLI Quick Configuration

To quickly configure this example, copy the following commands, paste them into a text file, remove any line breaks, change any details necessary to match your network configuration, and then copy and paste the commands into the CLI at the `[edit]` hierarchy level.

Device PE1

```
set interfaces ge-4/1/0 unit 0 family inet address 192.0.2.2/24
set interfaces ge-4/1/0 unit 0 description toPE2
set interfaces ge-0/2/0 unit 0 family inet address 10.10.10.1/30
set interfaces ge-0/2/0 unit 0 description toP1
set interfaces ge-0/2/0 unit 0 family mpls
set interfaces ge-0/2/4 unit 0 family inet address 10.10.15.2/30
set interfaces ge-0/2/4 unit 0 description toP5
set interfaces ge-0/2/4 unit 0 family mpls
set interfaces ge-4/1/0 unit 0 family inet address 192.0.2.2/24 vrrp-group 1 virtual-address
192.0.2.5
set interfaces ge-4/1/0 unit 0 family inet address 192.0.2.2/24 vrrp-group 1 priority 240
set interfaces ge-4/1/0 unit 0 family inet address 192.0.2.2/24 vrrp-group 1 fast-interval 100
set interfaces ge-4/1/0 unit 0 family inet address 192.0.2.2/24 vrrp-group 1 preempt
set interfaces ge-4/1/0 unit 0 family inet address 192.0.2.2/24 vrrp-group 1 accept-data
set interfaces lo0 unit 0 family inet address 10.255.8.207/32
set protocols mpls interface ge-0/2/0.0
set protocols mpls interface ge-0/2/4.0
set protocols bgp group pe-ce type internal
set protocols bgp group pe-ce local-address 10.255.8.207
set protocols bgp group pe-ce family inet-vpn unicast
set protocols bgp group pe-ce neighbor 10.255.8.86
set protocols bgp group pe-ce export send-routes
set protocols ospf area 0.0.0.0 interface ge-0/2/0.0
set protocols ospf area 0.0.0.0 interface ge-0/2/4.0
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set protocols ldp interface ge-0/2/0.0
set protocols ldp interface ge-0/2/4.0
set policy-options policy-statement send-routes term 1 from protocol direct
set policy-options policy-statement send-routes term 1 from protocol local
```

```
set policy-options policy-statement send-routes term 1 then accept
set routing-options nonstop-routing
set routing-options autonomous-system 100
set routing-instances cust1 instance-type vrf
set routing-instances cust1 interface ge-4/1/0.0
set routing-instances cust1 route-distinguisher 100:100
set routing-instances cust1 vrf-target target:100:100
set routing-instances cust1 vrf-table-label
set routing-instances cust1 routing-options interface ge-4/1/0.0 link-protection
```

Device PE2

```
set interfaces ge-0/0/2 unit 0 family inet address 10.10.12.2/30
set interfaces ge-0/0/2 unit 0 description toP2
set interfaces ge-0/0/2 unit 0 family mpls
set interfaces ge-0/1/2 unit 0 family inet address 10.10.13.1/30
set interfaces ge-0/1/2 unit 0 description toP4
set interfaces ge-0/1/2 unit 0 family mpls
set interfaces ge-2/0/2 unit 0 family inet address 192.0.2.3/24
set interfaces ge-2/0/2 unit 0 description toPE1
set interfaces ge-2/0/2 unit 0 family inet address 192.0.2.3/24 vrrp-group 1 virtual-address
192.0.2.5
set interfaces ge-2/0/2 unit 0 family inet address 192.0.2.3/24 vrrp-group 1 fast-interval 100
set interfaces ge-2/0/2 unit 0 family inet address 192.0.2.3/24 vrrp-group 1 preempt
set interfaces ge-2/0/2 unit 0 family inet address 192.0.2.3/24 vrrp-group 1 accept-data
set interfaces lo0 unit 0 family inet address 10.255.8.86/32
set protocols mpls interface ge-0/0/2.0
set protocols mpls interface ge-0/1/2.0
set protocols bgp group pe-ce type internal
set protocols bgp group pe-ce export send-routes
set protocols bgp group pe-ce local-address 10.255.8.86
set protocols bgp group pe-ce family inet-vpn unicast
set protocols bgp group pe-ce neighbor 10.255.8.207
set protocols ospf area 0.0.0.0 interface ge-0/0/2.0
set protocols ospf area 0.0.0.0 interface ge-0/1/2.0
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set protocols ldp interface ge-0/0/2.0
set protocols ldp interface ge-0/1/2.0
set policy-options policy-statement send-routes term 1 from protocol direct
set policy-options policy-statement send-routes term 1 from protocol local
set policy-options policy-statement send-routes term 1 then accept
set routing-options nonstop-routing
```

```
set routing-options autonomous-system 100
set routing-instances cust1 instance-type vrf
set routing-instances cust1 interface ge-2/0/2.0
set routing-instances cust1 route-distinguisher 100:100
set routing-instances cust1 vrf-target target:100:100
set routing-instances cust1 vrf-table-label
set routing-instances cust1 routing-options interface ge-2/0/2.0 link-protection
```

Device P1

```
set interfaces ge-0/0/3 unit 0 family inet address 10.10.11.1/30
set interfaces ge-0/0/3 unit 0 description toP2
set interfaces ge-0/0/3 unit 0 family mpls
set interfaces ge-0/0/4 unit 0 family inet address 10.10.10.2/30
set interfaces ge-0/0/4 unit 0 description toPE1
set interfaces ge-0/0/4 unit 0 family mpls
set protocols mpls interface ge-0/0/4.0
set protocols mpls interface ge-0/0/3.0
set protocols ospf area 0.0.0.0 interface ge-0/0/4.0
set protocols ospf area 0.0.0.0 interface ge-0/0/3.0
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set protocols ldp interface ge-0/0/3.0
set protocols ldp interface ge-0/0/4.0
set routing-options autonomous-system 100
```

Device P2

```
set interfaces ge-0/2/1 unit 0 family inet address 10.10.12.1/30
set interfaces ge-0/2/1 unit 0 description toPE2
set interfaces ge-0/2/1 unit 0 family mpls
set interfaces ge-1/2/1 unit 0 family inet address 10.10.11.2/30
set interfaces ge-1/2/1 unit 0 description toP1
set interfaces ge-1/2/1 unit 0 family mpls
set interfaces lo0 unit 0 family inet address 10.255.8.246/32
set protocols mpls interface all
set protocols mpls interface fxp0.0 disable
set protocols ospf area 0.0.0.0 interface all
set protocols ospf area 0.0.0.0 interface fxp0.0 disable
set protocols ldp interface all
set protocols ldp interface fxp0.0 disable
set routing-options autonomous-system 100
```

Device P4

```
set interfaces ge-0/2/3 unit 0 family inet address 10.10.13.2/30
set interfaces ge-0/2/3 unit 0 description toPE2
set interfaces ge-0/2/3 unit 0 family mpls
set interfaces ge-1/3/3 unit 0 family inet address 10.10.14.1/30
set interfaces ge-1/3/3 unit 0 description toP5
set interfaces ge-1/3/3 unit 0 family mpls
set interfaces lo0 unit 0 family inet address 10.255.8.4/32
set protocols mpls interface ge-0/2/3.0
set protocols mpls interface ge-1/3/3.0
set protocols ospf area 0.0.0.0 interface ge-0/2/3.0
set protocols ospf area 0.0.0.0 interface ge-1/3/3.0
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set protocols ldp interface ge-0/2/3.0
set protocols ldp interface ge-1/3/3.0
set routing-options autonomous-system 100
```

Device P5

```
set interfaces ge-0/1/2 unit 0 family inet address 10.10.15.1/30
set interfaces ge-0/1/2 unit 0 description toPE1
set interfaces ge-0/1/2 unit 0 family mpls
set interfaces ge-0/1/5 unit 0 family inet address 10.10.14.2/30
set interfaces ge-0/1/5 unit 0 description toP4
set interfaces ge-0/1/5 unit 0 family mpls
set interfaces lo0 unit 0 family inet address 10.255.8.5/32
set protocols mpls interface ge-0/1/5.0
set protocols mpls interface ge-0/1/2.0
set protocols ospf area 0.0.0.0 interface ge-0/1/5.0
set protocols ospf area 0.0.0.0 interface ge-0/1/2.0
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set protocols ldp interface ge-0/1/2.0
set protocols ldp interface ge-0/1/5.0
set routing-options autonomous-system 100
```

Procedure

Step-by-Step Procedure

The following example requires you to navigate various levels in the configuration hierarchy. For information about navigating the CLI, see *Using the CLI Editor in Configuration Mode* in the CLI User Guide.

To configure HFRR:

1. Configure the interfaces.

```
[edit interfaces]
user@PE1# set ge-4/1/0 unit 0 family inet address 192.0.2.2/24
user@PE1# set ge-4/1/0 unit 0 description toPE2
user@PE1# set ge-0/2/0 unit 0 family inet address 10.10.10.1/30
user@PE1# set ge-0/2/0 unit 0 description toP1
user@PE1# set ge-0/2/0 unit 0 family mpls
user@PE1# set ge-0/2/4 unit 0 family inet address 10.10.15.2/30
user@PE1# set ge-0/2/4 unit 0 description toP5
user@PE1# set ge-0/2/4 unit 0 family mpls
user@PE1# set lo0 unit 0 family inet address 10.255.8.207/32
```

2. (Optional) Configure VRRP on the interface to Device PE2.

```
[edit interfaces ge-4/1/0 unit 0 family inet address 192.0.2.2/24]
user@PE1# set vrrp-group 1 virtual-address 192.0.2.5
user@PE1# set vrrp-group 1 priority 240
user@PE1# set vrrp-group 1 fast-interval 100
user@PE1# set vrrp-group 1 preempt
user@PE1# set vrrp-group 1 accept-data
```

3. Configure MPLS on the interfaces.

```
[edit protocols mpls]
user@PE1# set interface ge-0/2/0.0
user@PE1# set interface ge-0/2/4.0
```

4. Configure BGP.

```
[edit protocols bgp group pe-ce]
user@PE1# set type internal
user@PE1# set local-address 10.255.8.207
user@PE1# set family inet-vpn unicast
user@PE1# set neighbor 10.255.8.86
user@PE1# set export send-routes
```

5. Configure a policy that advertises direct and local interface routes.

```
[edit policy-options policy-statement send-routes term 1]
user@PE1# set from protocol direct
user@PE1# set from protocol local
user@PE1# set then accept
```

6. Configure an interior gateway protocol, such as IS-IS or OSPF.

```
[edit protocols ospf area 0.0.0.0]
user@PE1# set interface ge-0/2/0.0
user@PE1# set interface ge-0/2/4.0
user@PE1# set interface lo0.0 passive
```

7. Configure a signaling protocol, such as RSVP or LDP.

```
[edit protocols ldp]
user@PE1# set interface ge-0/2/0.0
user@PE1# set interface ge-0/2/4.0
```

8. (Optional) Configure nonstop active routing.

```
[edit routing-options]
user@PE1# set nonstop-routing
```

9. Configure the autonomous system (AS).

```
[edit routing-options]
user@PE1# set routing-options autonomous-system 100
```

10. Configure the Layer 3 VPN routing instance.

```
[edit routing-instances cust1]
user@PE1# set instance-type vrf
user@PE1# set interface ge-4/1/0.0
user@PE1# set route-distinguisher 100:100
user@PE1# set vrf-target target:100:100
user@PE1# set vrf-table-label
```

11. Configure HFRR link protection.

```
[edit routing-instances cust1 routing-options]
user@PE1# set interface ge-4/1/0.0 link-protection (Host Fast Reroute)
```

12. If you are done configuring the device, commit the configuration.

```
[edit]
user@PE1# commit
```

Results

Confirm your configuration by issuing the `show interfaces`, `show protocols`, `show policy-options`, `show routing-options`, and `show routing-instances` commands.

```
user@PE1# show interfaces
ge-4/1/0 {
    unit 0 {
        description toPE2;
        family inet {
            address 192.0.2.2/24 {
                vrrp-group 1 {
                    virtual-address 192.0.2.5;
                    priority 240;
```

```
                    fast-interval 100;
                    preempt;
                    accept-data;
                }
            }
        }
    }
}
ge-0/2/0 {
    unit 0 {
        description toP1;
        family inet {
            address 10.10.10.1/30;
        }
        family mpls;
    }
}
ge-0/2/4 {
    unit 0 {
        description toP5;
        family inet {
            address 10.10.15.2/30;
        }
        family mpls;
    }
}
lo0 {
    unit 0 {
        family inet {
            address 10.255.8.207/32;
        }
    }
}
```

```
user@PE1# show protocols
mpls {
    interface ge-0/2/0.0;
    interface ge-0/2/4.0;
}
bgp {
    group pe-ce {
```

```
        export-send-routes;
        type internal;
        local-address 10.255.8.207;
        family inet-vpn {
            unicast;
        }
        neighbor 10.255.8.86;
    }
}
ospf {
    area 0.0.0.0 {
        interface ge-0/2/0.0;
        interface ge-0/2/4.0;
        interface lo0.0 {
            passive;
        }
    }
}
ldp {
    interface ge-0/2/0.0;
    interface ge-0/2/4.0;
}
```

```
user@PE1# show policy-options
policy-statement send-routes {
    term 1 {
        from protocol [ direct local ];
        then accept;
    }
}
```

```
user@PE1# show routing-options
nonstop-routing;
autonomous-system 100;
```

```
user@PE1# show routing-instances
cust1 {
    instance-type vrf;
    interface ge-4/1/0.0;
```

```
    route-distinguisher 100:100;
    vrf-target target:100:100;
    vrf-table-label;
    routing-options {
        interface {
            ge-4/1/0.0 {
                link-protection;
            }
        }
    }
}
```

Verification

IN THIS SECTION

Confirm that the configuration is working properly.

Verifying HFRR

Purpose

Make sure that HFRR is enabled.

Action

```
user@PE1> show hfrr profiles
HFRR pointer: 0x9250000
HFRR Current State: HFRR_ACTIVE
HFRR Protected IFL Name: ge-4/1/0.0
HFRR Protected IFL Handle: 0x921086c
HFRR Routing Instance Name: cust1
```

```
HFRR Routing Instance Handle: 0x9129740
HFRR Sync BG Sceduled: NO
HFRR RTS Filter On: YES
HFRR Delete BG Scheduled: NO
HFRR Num ARP Routes learnt: 100
HFRR Num FRR Routes Created: 100
```

Meaning

The output shows that the HFRR is enabled on interface ge-4/1/0.0.

Verifying ARP Routes

Purpose

Make sure that the expected ARP routes are learned.

Action

```
user@PE1> show route protocol arp
inet.0: 43 destinations, 43 routes (42 active, 0 holddown, 1 hidden)

inet.3: 3 destinations, 3 routes (3 active, 0 holddown, 0 hidden)

cust1.inet.0: 1033 destinations, 2043 routes (1033 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

192.0.2.3/24       @[ARP/4294967293] 00:04:35, from 192.0.2.1
                      Unusable
192.0.2.4/24       @[ARP/4294967293] 00:04:35, from 192.0.2.1
                      Unusable
192.0.2.5/24       @[ARP/4294967293] 00:04:32, from 192.0.2.1
                      Unusable
192.0.2.6/24       @[ARP/4294967293] 00:04:34, from 192.0.2.1
                      Unusable
192.0.2.7/24       @[ARP/4294967293] 00:04:35, from 192.0.2.1
                      Unusable
192.0.2.8/24       @[ARP/4294967293] 00:04:35, from 192.0.2.1
                      Unusable
192.0.2.9/24       @[ARP/4294967293] 00:04:35, from 192.0.2.1
                      Unusable
```

```
192.0.2.10/24       @[ARP/4294967293] 00:04:35, from 192.0.2.1
                      Unusable
192.0.2.11/24       @[ARP/4294967293] 00:04:33, from 192.0.2.1
                      Unusable
192.0.2.12/24       @[ARP/4294967293] 00:04:33, from 192.0.2.1
                      Unusable
192.0.2.13/24       @[ARP/4294967293] 00:04:33, from 192.0.2.1
                      Unusable
...
```

Verifying Fast Reroute Routes

Purpose

Make sure that the expected fast reroute (FRR) routes are learned.

Action

```
user@PE1> show route protocol frr
inet.0: 43 destinations, 43 routes (42 active, 0 holddown, 1 hidden)

inet.3: 3 destinations, 3 routes (3 active, 0 holddown, 0 hidden)

cust1.inet.0: 1033 destinations, 2043 routes (1033 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

192.0.2.3/24       #[FRR/200] 00:05:38, from 192.0.2.1
                    > to 192.0.2.3 via ge-4/1/0.0
                      to 10.10.15.1 via ge-0/2/4.0, Push 16, Push 299792(top)
192.0.2.4/24       #[FRR/200] 00:05:38, from 192.0.2.1
                    > to 192.0.2.4 via ge-4/1/0.0
                      to 10.10.15.1 via ge-0/2/4.0, Push 16, Push 299792(top)
192.0.2.5/24       #[FRR/200] 00:05:35, from 192.0.2.1
                    > to 192.0.2.5 via ge-4/1/0.0
                      to 10.10.15.1 via ge-0/2/4.0, Push 16, Push 299792(top)
192.0.2.6/24       #[FRR/200] 00:05:37, from 192.0.2.1
                    > to 192.0.2.6 via ge-4/1/0.0
                      to 10.10.15.1 via ge-0/2/4.0, Push 16, Push 299792(top)
192.0.2.7/24       #[FRR/200] 00:05:38, from 192.0.2.1
                    > to 192.0.2.7 via ge-4/1/0.0
                      to 10.10.15.1 via ge-0/2/4.0, Push 16, Push 299792(top)
```

```
192.0.2.8/24       #[FRR/200] 00:05:38, from 192.0.2.1
                    > to 192.0.2.8 via ge-4/1/0.0
                      to 10.10.15.1 via ge-0/2/4.0, Push 16, Push 299792(top)
192.0.2.9/24       #[FRR/200] 00:05:38, from 192.0.2.1
                    > to 192.0.2.9 via ge-4/1/0.0
                      to 10.10.15.1 via ge-0/2/4.0, Push 16, Push 299792(top)
192.0.2.10/24      #[FRR/200] 00:05:38, from 192.0.2.1
...
```

Verifying Forwarding

Purpose

Make sure that the expected routes appear in the forwarding table.

Action

```
user@PE1> show route forwarding-table destination 192.0.2.3
Routing table: default.inet
Internet:
Destination        Type RtRef Next hop           Type Index NhRef Netif
default            perm     0                    rjct    36     1

Routing table: default-switch.inet
Internet:
Destination        Type RtRef Next hop           Type Index NhRef Netif
default            perm     0                    rjct   554     1

Routing table: __master.anon__.inet
Internet:
Destination        Type RtRef Next hop           Type Index NhRef Netif
default            perm     0                    rjct   532     1

Routing table: cust1.inet
Internet:
Destination        Type RtRef Next hop           Type Index NhRef Netif
192.0.2.3/24       user     0                    ulst 1048575     2
                              0:0:14:14:1:3      ucst   767     3 ge-4/1/0.0
                                                 indr 1048574  1001
                              10.10.15.1        Push 16, Push 299792(top)  1262     2 ge-0/2/4.0
192.0.2.3/24       dest     0 0:0:14:14:1:3      ucst   767     3 ge-4/1/0.0
```

```
...
```

Unicast Reverse Path Forwarding Check for VPNs

IN THIS SECTION

Understanding Unicast RPF (Switches)

IN THIS SECTION

To protect against IP spoofing, and some types of denial-of-service (DoS) and distributed denial-of-service (DDoS) attacks, unicast reverse-path-forwarding (RPF) verifies that packets are arriving from a legitimate path. It does this by checking the source address of each packet that arrives on an untrusted ingress interface and, comparing it to the forwarding-table entry for its source address. If the packet is from a valid path, that is, one that the sender would use to reach the destination, the device forwards the packet to the destination address. If it is not from a valid path, the device discards the packet. Unless it is protected against, IP spoofing can be an effective way for intruders to pass IP packets to a destination as genuine traffic, when in fact the packets are not actually meant for the destination.

Unicast RPF is supported for the IPv4 and IPv6 protocol families, as well as for the virtual private network (VPN) address family. Unicast RPF is not supported on interfaces configured as tunnel sources. This affects only the transit packets exiting the tunnel.

NOTE: RPF check is not supported on vxlan-enabled interface on QFX Series and EX Series switches.

There are two modes of unicast RPF, *strict mode*, and *loose mode*. The default is strict mode, which means the switch forwards a packet only if the receiving interface is the best return path to the packet's unicast source address. Strict mode is especially useful on untrusted interfaces (where untrusted users or processes can place packets on the network segment), and for symmetrically routed interfaces (see "When to Enable Unicast RPF" on page 1134.) For more information about strict unicast RPF, see RFC 3704, *Ingress Filtering for Multihomed Networks* at http://www.ietf.org/rfc/rfc3704.txt.

To enable strict mode unicast RPF on a selected customer-edge interface:

[edit interfaces]user@switch# **set *interface-name* unit 0 family inet rpf-check**

The other mode is loose mode, which means the system checks to see if the packet has a source address with a corresponding prefix in the routing table, but it does not check whether the receiving interface is the best return path to the packet's unicast source address.

To enable unicast RPF loose mode, enter:

[edit interfaces]user@switch# **set *interface-name* unit 0 family inet rpf-check mode loose**

NOTE:
globally"Limitations of the Unicast RPF Implementation on EX3200, EX4200, and EX4300 Switches" on page 1136

Unicast RPF for Switches Overview

Unicast RPF functions as an ingress filter that reduces the forwarding of IP packets that might be spoofing an address. By default, unicast RPF is disabled on the switch interfaces. The switch supports only the active paths method of determining the best return path back to a unicast source address. The active paths method looks up the best reverse path entry in the forwarding table. It does not consider alternate routes specified using routing-protocol-specific methods when determining the best return path.

If the forwarding table lists the receiving interface as the interface to use to forward the packet back to its unicast source, it is the best return path interface.

Unicast RPF Implementation

Unicast RPF Packet Filtering

When you enable unicast RPF on the switch, the switch handles traffic in the following manner:

- If the switch receives a packet on the interface that is the best return path to the unicast source address of that packet, the switch forwards the packet.
- If the best return path from the switch to the packet's unicast source address is not the receiving interface, the switch discards the packet.
- If the switch receives a packet that has a source IP address that does not have a routing entry in the forwarding table, the switch discards the packet.

Bootstrap Protocol (BOOTP) and DHCP Requests

Bootstrap protocol (BOOTP) and DHCP request packets are sent with a broadcast MAC address and therefore the switch does not perform unicast RPF checks on them. The switch forwards all BOOTP packets and DHCP request packets without performing unicast RPF checks.

Default Route Handling

If the best return path to the source is the default route (`0.0.0.0`) and the default route points to `reject`, the switch discards the packets. If the default route points to a valid network interface, the switch performs a normal unicast RPF check on the packets.

NOTE: On the EX4300, the default route is not used when the switch is configured in unicast RPF strict mode.

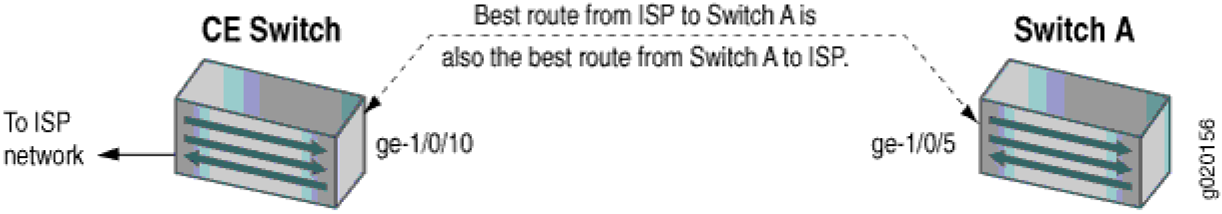
When to Enable Unicast RPF

Enable unicast RPF when you want to ensure that traffic arriving on a network interface comes from a source that resides on a network that interface can reach. You can enable unicast RPF on untrusted interfaces to filter spoofed packets. For example, a common application for unicast RPF is to help defend an enterprise network from DoS/DDoS attacks coming from the Internet.

Enable unicast RPF only on symmetrically routed interfaces, and as close as possible to the traffic source stops spoofed traffic before it can proliferate or reach interfaces that do not have unicast RPF enabled. Because unicast RPF is enabled globally on EX3200, EX4200, and EX4300 switches, ensure that *all* interfaces are symmetrically routed before you enable unicast RPF on these switches, as shown in Figure 87 on page 1135. Enabling unicast RPF on asymmetrically routed interfaces results in packets from legitimate sources being filtered. A symmetrically routed interface uses the same route in both directions between the source and the destination.

Unicast RPF is enabled globally on EX3200, EX4200, and EX4300 switches, to with these devices, be sure that *all* interfaces are symmetrically routed before you enable unicast RPF on these switches. Enabling unicast RPF on asymmetrically routed interfaces results in packets from legitimate sources being filtered.

Figure 87: Symmetrically Routed Interfaces



The following switch interfaces are most likely to be symmetrically routed and thus are candidates for unicast RPF enabling:

- The service provider edge to a customer
- The customer edge to a service provider
- A single access point out of the network (usually on the network perimeter)
- A terminal network that has only one link

On EX3200, EX4200, and EX4300 switches, we recommend that you enable unicast RPF explicitly on either all interfaces or only one interface. To avoid possible confusion, do not enable it on only some interfaces:

- Enabling unicast RPF explicitly on only one interface makes it easier if you choose to disable it in the future because you must explicitly disable unicast RPF on every interface on which you explicitly enabled it. If you explicitly enable unicast RPF on two interfaces and you disable it on only one interface, unicast RPF is still implicitly enabled globally on the switch. The drawback of this approach is that the switch displays the flag that indicates that unicast RPF is enabled only on interfaces on which unicast RPF is explicitly enabled, so even though unicast RPF is enabled on all interfaces, this status is not displayed.
- Enabling unicast RPF explicitly on all interfaces makes it easier to know whether unicast RPF is enabled on the switch because every interface shows the correct status. (Only interfaces on which you explicitly enable unicast RPF display the flag that indicates that unicast RPF is enabled.) The drawback of this approach is that if you want to disable unicast RPF, you must explicitly disable it on every interface. If unicast RPF is enabled on any interface, it is implicitly enabled on all interfaces.

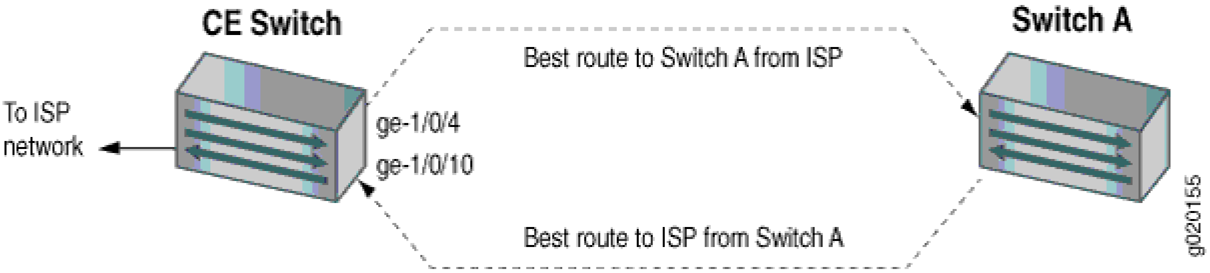
When Not to Enable Unicast RPF

Typically, you will not enable unicast RPF if:

- Switch interfaces are multihomed.
- Switch interfaces are trusted interfaces.
- BGP is carrying prefixes and some of those prefixes are not advertised or are not accepted by the ISP under its policy. (The effect in this case is the same as filtering an interface by using an incomplete access list.)
- Switch interfaces face the network core. Core-facing interfaces are usually asymmetrically routed.

An asymmetrically routed interface uses different paths to send and receive packets between the source and the destination, as shown in Figure 88 on page 1136. This means that if an interface receives a packet, that interface does not match the forwarding table entry as the best return path back to the source. If the receiving interface is not the best return path to the source of a packet, unicast RPF causes the switch to discard the packet even though it comes from a valid source.

Figure 88: Asymmetrically Routed Interfaces



> **NOTE**: Do not enable unicast RPF on EX3200, EX4200, and EX4300 switches if any switch interfaces are asymmetrically routed, because unicast RPF is enabled globally on all interfaces of these switches. All switch interfaces must be symmetrically routed for you to enable unicast RPF without the risk of the switch discarding traffic that you want to forward.

Limitations of the Unicast RPF Implementation on EX3200, EX4200, and EX4300 Switches

On EX3200, EX4200, and EX4300 switches, the switch implements unicast RPF on a global basis. You cannot enable unicast RPF on a per-interface basis. Unicast RPF is globally disabled by default.

- When you enable unicast RPF on any interface, it is automatically enabled on all switch interfaces, including link aggregation groups (LAGs), integrated routing and bridging (IRB) interfaces, and routed VLAN interfaces (RVIs).
- When you disable unicast RPF on the interface (or interfaces) on which you enabled unicast RPF, it is automatically disabled on all switch interfaces.

NOTE: You must explicitly disable unicast RPF on every interface on which it was explicitly enabled or unicast RPF remains enabled on all switch interfaces.

QFX switches, OCX switches, and EX3200 and EX4200 switches do not perform unicast RPF filtering on equal-cost multipath (ECMP) traffic. The unicast RPF check examines only one best return path to the packet source, but ECMP traffic employs an address block consisting of multiple paths. Using unicast RPF to filter ECMP traffic on these switches can result in the switch discarding packets that you want to forward because the unicast RPF filter does not examine the entire ECMP address block.

SEE ALSO

Example: Configuring Unicast RPF (On a Switch)
Troubleshooting Unicast RPF

Example: Configuring Unicast RPF (On a Router)

IN THIS SECTION

This example shows how to help defend ingress interfaces against denial-of-service (DoS) and distributed denial-of-service (DDoS) attacks by configuring unicast RPF on a customer-edge interface to filter incoming traffic.

Requirements

No special configuration beyond device initialization is required.

Overview

IN THIS SECTION

- Topology | **1138**

In this example, Device A is using OSPF to advertise a prefix for the link that connects to Device D. Device B has unicast RPF configured. OSPF is enabled on the links between Device B and Device C and the links between Device A and Device C, but not on the links between Device A and Device B. Therefore, Device B learns about the route to Device D through Device C.

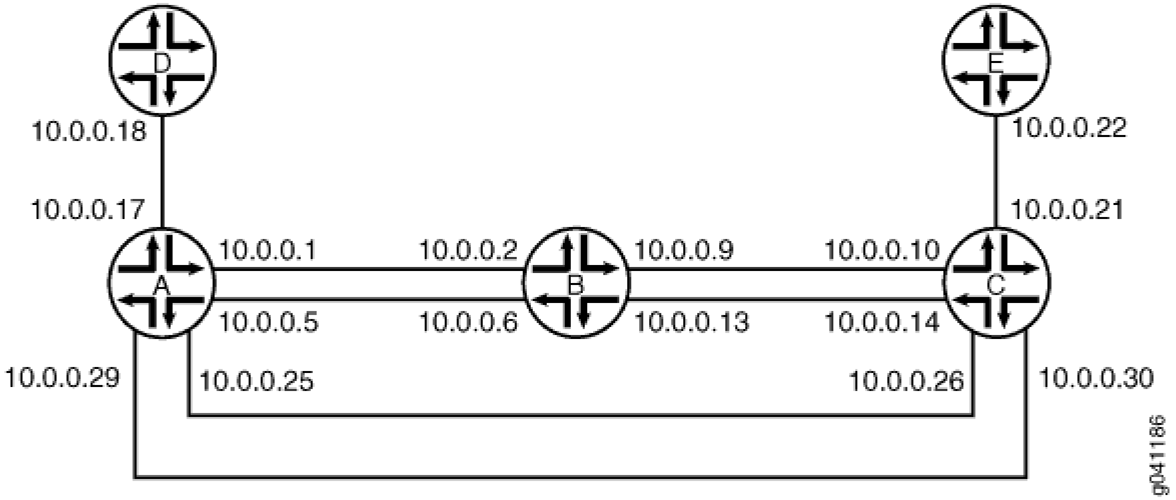
If ingress filtering is used in an environment where DHCP or BOOTP is used, it should be ensured that the packets with a source address of 0.0.0.0 and a destination address of 255.255.255.255 are allowed to reach the relay agent in routers when appropriate.

This example also includes a fail filter. When a packet fails the unicast RPF check, the fail filter is evaluated to determine if the packet should be accepted anyway. The fail filter in this example allows Device B's interfaces to accept Dynamic Host Configuration Protocol (DHCP) packets. The filter accepts all packets with a source address of 0.0.0.0 and a destination address of 255.255.255.255.

Topology

Figure 89 on page 1138 shows the sample network.

Figure 89: Unicast RPF Sample Topoolgy



Configuration

IN THIS SECTION

CLI Quick Configuration

To quickly configure this example, copy the following commands, paste them into a text file, remove any line breaks, change any details necessary to match your network configuration, and then copy and paste the commands into the CLI at the `[edit]` hierarchy level.

Device A

```
set interfaces fe-1/2/0 unit 1 family inet address 10.0.0.1/30
set interfaces fe-0/0/2 unit 5 family inet address 10.0.0.5/30
set interfaces fe-0/0/1 unit 17 family inet address 10.0.0.17/30
set interfaces fe-0/1/1 unit 25 family inet address 10.0.0.25/30
set interfaces fe-1/1/1 unit 29 family inet address 10.0.0.29/30
set protocols ospf export send-direct
set protocols ospf area 0.0.0.0 interface fe-0/1/1.25
set protocols ospf area 0.0.0.0 interface fe-1/1/1.29
set policy-options policy-statement send-direct from protocol direct
set policy-options policy-statement send-direct from route-filter 10.0.0.16/30 exact
set policy-options policy-statement send-direct then accept
```

Device B

```
set interfaces fe-1/2/0 unit 2 family inet rpf-check fail-filter rpf-special-case-dhcp
set interfaces fe-1/2/0 unit 2 family inet address 10.0.0.2/30
set interfaces fe-1/1/1 unit 6 family inet rpf-check fail-filter rpf-special-case-dhcp
set interfaces fe-1/1/1 unit 6 family inet address 10.0.0.6/30
set interfaces fe-0/1/1 unit 9 family inet rpf-check fail-filter rpf-special-case-dhcp
set interfaces fe-0/1/1 unit 9 family inet address 10.0.0.9/30
set interfaces fe-0/1/0 unit 13 family inet rpf-check fail-filter rpf-special-case-dhcp
```

```
set interfaces fe-0/1/0 unit 13 family inet address 10.0.0.13/30
set protocols ospf area 0.0.0.0 interface fe-0/1/1.9
set protocols ospf area 0.0.0.0 interface fe-0/1/0.13
set routing-options forwarding-table unicast-reverse-path active-paths
set firewall filter rpf-special-case-dhcp term allow-dhcp from source-address 0.0.0.0/32
set firewall filter rpf-special-case-dhcp term allow-dhcp from destination-address
255.255.255.255/32
set firewall filter rpf-special-case-dhcp term allow-dhcp then count rpf-dhcp-traffic
set firewall filter rpf-special-case-dhcp term allow-dhcp then accept
set firewall filter rpf-special-case-dhcp term default then log
set firewall filter rpf-special-case-dhcp term default then reject
```

Device C

```
set interfaces fe-1/2/0 unit 10 family inet address 10.0.0.10/30
set interfaces fe-0/0/2 unit 14 family inet address 10.0.0.14/30
set interfaces fe-1/0/2 unit 21 family inet address 10.0.0.21/30
set interfaces fe-1/2/2 unit 26 family inet address 10.0.0.26/30
set interfaces fe-1/2/1 unit 30 family inet address 10.0.0.30/30
set protocols ospf area 0.0.0.0 interface fe-1/2/0.10
set protocols ospf area 0.0.0.0 interface fe-0/0/2.14
set protocols ospf area 0.0.0.0 interface fe-1/2/2.26
set protocols ospf area 0.0.0.0 interface fe-1/2/1.30
```

Device D

```
set interfaces fe-1/2/0 unit 18 family inet address 10.0.0.18/30
```

Device E

```
set interfaces fe-1/2/0 unit 22 family inet address 10.0.0.22/30
```

Configuring Device A

Step-by-Step Procedure

The following example requires you to navigate various levels in the configuration hierarchy. For information about navigating the CLI, see *Using the CLI Editor in Configuration Mode*.

To configure Device A:

1. Configure the interfaces.

   ```
   [edit interfaces]
   user@A# set fe-1/2/0 unit 1 family inet address 10.0.0.1/30
   user@A# set fe-0/0/2 unit 5 family inet address 10.0.0.5/30
   user@A# set fe-0/0/1 unit 17 family inet address 10.0.0.17/30
   user@A# set fe-0/1/1 unit 25 family inet address 10.0.0.25/30
   user@A# set fe-1/1/1 unit 29 family inet address 10.0.0.29/30
   ```

2. Configure OSPF.

   ```
   [edit protocols ospf]
   user@A# set export send-direct
   user@A# set area 0.0.0.0 interface fe-0/1/1.25
   user@A# set area 0.0.0.0 interface fe-1/1/1.29
   ```

3. Configure the routing policy.

   ```
   [edit policy-options policy-statement send-direct]
   user@A# set from protocol direct
   user@A# set from route-filter 10.0.0.16/30 exact
   user@A# set then accept
   ```

4. If you are done configuring Device A, commit the configuration.

   ```
   [edit]
   user@A# commit
   ```

Configuring Device B

Step-by-Step Procedure

The following example requires you to navigate various levels in the configuration hierarchy. For information about navigating the CLI, see *Using the CLI Editor in Configuration Mode*.

To configure Device B:

1. Configure the interfaces.

```
[edit interfaces]
user@B# set fe-1/2/0 unit 2 family inet address 10.0.0.2/30
user@B# set fe-1/1/1 unit 6 family inet address 10.0.0.6/30
user@B# set fe-0/1/1 unit 9 family inet address 10.0.0.9/30
user@B# set fe-0/1/0 unit 13 family inet address 10.0.0.13/30
```

2. Configure OSPF.

```
[edit protocols ospf area 0.0.0.0]
user@B# set interface fe-0/1/1.9
user@B# set interface fe-0/1/0.13
```

3. Configure unicast RPF, and apply the optional fail filter.

```
[edit interfaces]
user@B# set fe-1/2/0 unit 2 family inet rpf-check fail-filter rpf-special-case-dhcp
user@B# set fe-1/1/1 unit 6 family inet rpf-check fail-filter rpf-special-case-dhcp
user@B# set fe-0/1/1 unit 9 family inet rpf-check fail-filter rpf-special-case-dhcp
user@B# set fe-0/1/0 unit 13 family inet rpf-check fail-filter rpf-special-case-dhcp
```

4. (Optional) Configure the fail filter that gets evaluated if a packet fails the RPF check.

```
[edit firewall filter rpf-special-case-dhcp]
user@B# set term allow-dhcp from source-address 0.0.0.0/32
user@B# set term allow-dhcp from destination-address 255.255.255.255/32
user@B# set term allow-dhcp then count rpf-dhcp-traffic
user@B# set term allow-dhcp then accept
user@B# set term default then log
user@B# set term default then reject
```

5. (Optional) Configure only active paths to be considered in the RPF check.

   This is the default behavior.

```
[edit routing-options forwarding-table]
user@B# set unicast-reverse-path active-paths
```

6. If you are done configuring Device B, commit the configuration.

```
[edit]
user@B# commit
```

Results

Confirm your configuration by issuing the `show firewall`, `show interfaces`, `show protocols`, `show routing-options`, and `show policy-options` commands. If the output does not display the intended configuration, repeat the instructions in this example to correct the configuration.

Device A

```
user@A# show interfaces
fe-1/2/0 {
    unit 1 {
        family inet {
            address 10.0.0.1/30;
        }
    }
}
fe-0/0/2 {
    unit 5 {
        family inet {
            address 10.0.0.5/30;
        }
    }
}
fe-0/0/1 {
    unit 17 {
        family inet {
            address 10.0.0.17/30;
        }
    }
}
fe-0/1/1 {
    unit 25 {
        family inet {
            address 10.0.0.25/30;
        }
    }
```

```
}
fe-1/1/1 {
    unit 29 {
        family inet {
            address 10.0.0.29/30;
        }
    }
}
```

```
user@A# show protocols
ospf {
    export send-direct;
    area 0.0.0.0 {
        interface fe-0/1/1.25;
        interface fe-1/1/1.29;
    }
}
```

```
user@A# show policy-options
policy-statement send-direct {
    from {
        protocol direct;
        route-filter 10.0.0.16/30 exact;
    }
    then accept;
}
```

Device B

```
user@B# show firewall
filter rpf-special-case-dhcp {
    term allow-dhcp {
        from {
            source-address {
                0.0.0.0/32;
            }
            destination-address {
                255.255.255.255/32;
            }
```

```
        }
        then {
            count rpf-dhcp-traffic;
            accept;
        }
    }
    term default {
        then {
            log;
            reject;
        }
    }
}
user@B# show interfaces
fe-1/2/0 {
    unit 2 {
        family inet {
            rpf-check fail-filter rpf-special-case-dhcp;
            address 10.0.0.2/30;
        }
    }
}
fe-1/1/1 {
    unit 6 {
        family inet {
            rpf-check fail-filter rpf-special-case-dhcp;
            address 10.0.0.6/30;
        }
    }
}
fe-0/1/1 {
    unit 9 {
        family inet {
            rpf-check fail-filter rpf-special-case-dhcp;
            address 10.0.0.9/30;
        }
    }
}
fe-0/1/0 {
    unit 13 {
        family inet {
            rpf-check fail-filter rpf-special-case-dhcp;
            address 10.0.0.13/30;
```

```
        }
    }
}
```

```
user@B# show protocols
ospf {
    area 0.0.0.0 {
        interface fe-0/1/1.9;
        interface fe-0/1/0.13;
    }
}
```

```
user@B# show routing-options
forwarding-table {
    unicast-reverse-path active-paths;
}
```

Enter the configurations on Device C, Device D, and Device E, as shown in "CLI Quick Configuration" on page 1139.

Verification

IN THIS SECTION

Confirm that the configuration is working properly.

Confirm That Unicast RPF Is Enabled

Purpose

Make sure that the interfaces on Device B have unicast RPF enabled.

Action

```
user@B> show interfaces fe-0/1/0.13 extensive
 Logical interface fe-0/1/0.13 (Index 73) (SNMP ifIndex 553) (Generation 208)
    Flags: SNMP-Traps 0x4000 Encapsulation: ENET2
    Traffic statistics:
     Input  bytes  :               999390
     Output bytes  :              1230122
     Input  packets:                12563
     Output packets:                12613
    Local statistics:
     Input  bytes  :               998994
     Output bytes  :              1230122
     Input  packets:                12563
     Output packets:                12613
    Transit statistics:
     Input  bytes  :                  396                    0 bps
     Output bytes  :                    0                    0 bps
     Input  packets:                    0                    0 pps
     Output packets:                    0                    0 pps
    Protocol inet, MTU: 1500, Generation: 289, Route table: 22
      Flags: Sendbcast-pkt-to-re, uRPF
      RPF Failures: Packets: 0, Bytes: 0
      Addresses, Flags: Is-Preferred Is-Primary
        Destination: 10.0.0.12/30, Local: 10.0.0.13, Broadcast: 10.0.0.15, Generation: 241
```

Meaning

The **uRPF** flag confirms that unicast RPF is enabled on this interface.

Confirm That the Source Addresses Are Blocked

Purpose

Use the `ping` command to make sure that Device B blocks traffic from unexpected source addresses.

Action

From Device A, ping Device B’s interfaces, using 10.0.0.17 as the source address.

```
user@A> ping 10.0.0.6 source 10.0.0.17
PING 10.0.0.6 (10.0.0.6): 56 data bytes
^C
--- 10.0.0.6 ping statistics ---
3 packets transmitted, 0 packets received, 100% packet loss
```

Meaning

As expected, the ping operation fails.

Confirm That the Source Addresses Are Unblocked

Purpose

Use the `ping` command to make sure that Device B does not block traffic when the RPF check is deactivated.

Action

1. Deactivate the RPF check on one of the interfaces.
2. Rerun the ping operation.

```
user@B> deactivate interfaces fe-1/1/1.6 family inet rpf-check

user@A> ping 10.0.0.6 source 10.0.0.17
PING 10.0.0.2 (10.0.0.2): 56 data bytes
64 bytes from 10.0.0.2: icmp_seq=0 ttl=63 time=1.316 ms
64 bytes from 10.0.0.2: icmp_seq=1 ttl=63 time=1.263 ms
^C
--- 10.0.0.2 ping statistics ---
2 packets transmitted, 2 packets received, 0% packet loss
round-trip min/avg/max/stddev = 1.263/1.289/1.316/0.027 ms
```

Meaning

As expected, the ping operation succeeds.

RELATED DOCUMENTATION

Example: Configuring Unicast RPF (On a Switch)

Troubleshooting Unicast RPF

Load Balancing in Layer 3 VPNs

IN THIS SECTION

VPN Per-Packet Load Balancing

By default, when there are multiple equal-cost paths to the same destination for the active route, the Junos OS software uses a hash algorithm to select one of the next-hop addresses to install in the forwarding table. Whenever the set of next hops for a destination changes, this selection process (using the same hash algorithm) is repeated to choose the best single next-hop address using the same hash algorithm.

Alternatively, you can configure the Junos OS software to spread the VPN traffic across the multiple valid paths between PE devices. This feature is called per-packet load balancing. VPN traffic load balancing is only possible when more than one valid path is available. You can configure Junos OS so that, for the active route, all next-hop addresses for a destination are installed in the forwarding table. In addition to increasing the volume of traffic you can send between VPN devices, you can configure per-packet load balancing to optimize traffic flows across multiple paths.

Traffic is distributed across multiple valid paths by running a hash algorithm on various elements of the route, such as the MPLS label or the destination address. The following tables describe how the load balancing hash algorithm is run on routes at the ingress router and at the transit and egress routers. The route elements used by the hash algorithm vary depending on VPN application. If Junos OS encounters an S-bit set to 1 (indicating the bottom of the stack), it does not apply the hash algorithm any further.

Table 9: Ingress Router Hashing

Application	Ingress *Logical Interface*	MPLS Labels	Source and Destination MAC Addresses	Reordering and Flow Separation Risk	Disable Control Word	IP (Source/ Destination Address and Port, Protocol)
Layer 2 VPNs and Layer 2 Circuits configured with CCC	Yes	Yes	No	Yes (if the data is variable, for example ATM)	Yes	N/A
Layer 2 VPNs and Layer 2 Circuits configured with TCC	Yes	Yes	No	Yes (if the data is variable, for example ATM)	Yes	N/A
Layer 3 VPNs and IPv4 or IPv6 RIBs	Yes	No	No	No	No	Yes
VPLS	Yes	No	Yes	No	No	Yes

Table 10: Transit and Egress Router Hashing

Application	Ingress Logical Interface	MPLS Labels (up to 3 and the S-bit is set to 1)	Reordering and Flow Separation Risk	IP (Source/ Destination Address and Port, Protocol)
Layer 2 VPNs and Layer 2 Circuits configured with CCC	Yes	Yes	No	No

Table 10: Transit and Egress Router Hashing *(Continued)*

Application	Ingress Logical Interface	MPLS Labels (up to 3 and the S-bit is set to 1)	Reordering and Flow Separation Risk	IP (Source/ Destination Address and Port, Protocol)
Layer 2 VPNs and Layer 2 Circuits configured with TCC	Yes	Yes	No	Yes
Layer 3 VPNs and IPv4 or IPv6 RIBs	Yes	Yes	No	Yes
VPLS	Yes	Yes for known unicast traffic No for broadcast, unicast unknown, and multicast traffic	No	No

Load Balancing and IP Header Filtering for Layer 3 VPNs

You can now simultaneously enable both load balancing of traffic across both internal and external BGP paths and filtering of traffic based on the IP header. This enables you to configure filters and policers at the egress PE router for traffic that is simultaneously being load-balanced across both internal and external BGP paths. This feature is available only on the M120 router, M320 router, MX Series routers, and T Series routers.

To enable these features on a Layer 3 VPN routing instance, include the `vpn-unequal-cost equal-external-internal` statement at the `[edit routing-instances` *`routing-instance-name`* `routing-options multipath]` hierarchy level and the `vrf-table-label` statement at the `[edit routing-instances` *`routing-instance-name`*`]` hierarchy level.

If you issue the `show route detail` command, you can discover whether or not a route is being load-balanced (equal-external-internal) and what its interface index is.

If you have also configured fast reroute, please be aware of the following behavior:

- If an IBGP path goes down, it could be replaced by either an active EBGP path or an active IBGP path.
- If an EBGP path goes down, it can only be replaced by another active EBGP path. This prevents the forwarding of core-facing interface traffic to an IBGP destination.

NOTE: You can include the `vpn-unequal-cost equal-external-internal` statement and the `l3vpn` statement at the `[edit routing-options forwarding-options chained-composite-next-hop ingress]` hierarchy level simultaneously. However, if you do this, EBGP does not work. This means that when there are both paths with chained nexthops and paths with nonchained nexthops as candidates for EBGP equal-cost multipath (ECMP), the paths using chained nexthops are excluded. In a typical case, the excluded paths are the internal paths.

Layer 3 VPN Load Balancing Overview

The load balancing feature allows a device to divide incoming and outgoing traffic along multiple paths in order to reduce congestion in the network. Load balancing improves the utilization of various network paths, and provides more effective network bandwidth.

When multiple protocols are in use, the device uses the *route preference* value (also known as the *administrative distance* value) to select a route. While using a single routing protocol, the router chooses the path with the lowest cost (or metric) to the destination. If the device receives and installs multiple paths with the same route preference and same cost to a destination, load balancing must be configured.

In a network with both internal and external BGP paths installed among devices in different autonomous systems, BGP selects only a single best path by default, and does not perform load balancing. A Layer 3 VPN with internal and external BGP paths uses the `multipath` statement for protocol-independent load balancing. When you include the `multipath` statement in a routing instance, protocol-independent load balancing is applied to the default routing table for that routing instance. By using the `vpn-unequal-cost` statement, protocol-independent load balancing is applied to VPN routes. By using the `equal-external-internal` statement, protocol-independent load balancing is applied to both internal and external BGP paths and can be configured in conjunction with IP header filtering (enabled with the `vrf-table-label` statement).

Example: Load Balancing Layer 3 VPN Traffic While Simultaneously Using IP Header Filtering

IN THIS SECTION

NOTE: Our content testing team has validated and updated this example.

This example shows how to configure load balancing in a Layer 3 VPN (with internal and external BGP paths) while simultaneously using IP header filtering.

Requirements

This example requires the following hardware and software components:

- M Series Multiservice Edge Routers (M120 and M320 only), MX Series 5G Universal Routing Platforms, T Series Core Routers, or PTX Series Transport Routers.
- Junos OS Release 12.1 or later
  - Revalidated on Junos OS Release 20.1R1 for MX Series routers

Overview

IN THIS SECTION

The following example shows how to configure load balancing while simultaneously using IP header filtering in a Layer 3 VPN.

NOTE: This example demonstrates how load balancing and IP header filtering work together. The testing of IP header filtering is out of the scope of this example.

The Junos OS BGP provides a multipath feature that allows load balancing between peers in the same or different autonomous systems (ASs). This example uses the `equal-external-internal` statement at the `[edit routing-instances instance-name routing-options multipath vpn-unequal-cost]` hierarchy level to perform load balancing. The `vrf-table-label` statement is configured at the `[edit routing-instances instance-name]` hierarchy level to enable IP header filtering.

```
[edit]
routing-instances {
    instance-name {
```

```
        vrf-table-label;
        routing-options {
            multipath {
                vpn-unequal-cost {
                    equal-external-internal;
                }
            }
        }
    }
}
```

NOTE: These statements are available only in the context of a routing instance.

In this example, Device CE1 is in AS1 and connected to Device PE1. Devices PE1, PE2, PE3, and P are in AS2. Device CE2 is connected to Devices PE2 and PE3 and is in AS3. Device CE3 is connected to Device PE3 and is in AS4. BGP and MPLS are configured through the network. OSPF is the interior gateway protocol (IGP) that is used in this network.

The configuration for Devices PE1, PE2, and PE3 includes the `equal-external-internal` statement at the `[edit routing-instances instance-name routing-options multipath vpn-unequal-cost]` hierarchy level to enable load balancing in the network. IP header filtering is enabled when the `vrf-table-label` statement is configured at the `[edit routing-instances instance-name]` hierarchy level on the PE devices.

Figure 90 on page 1155 shows the topology used in this example.

Topology

Figure 90: Layer 3 VPN Load Balancing Using IP Header Filtering

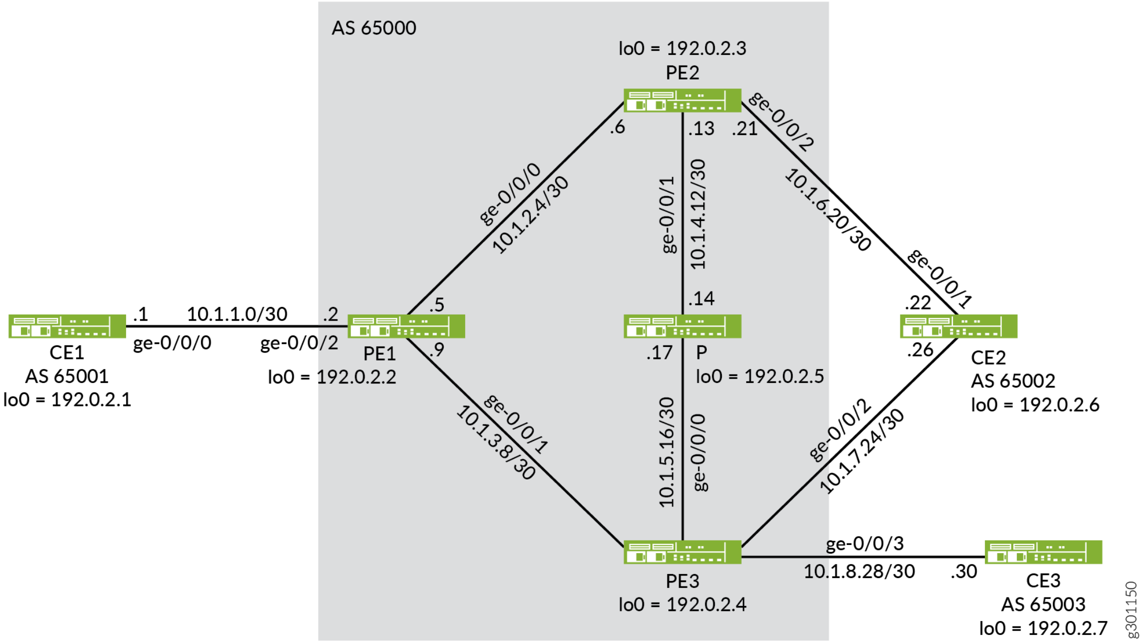


Table 11 on page 1155 shows the list of IP addresses used in this example for quick reference.

Table 11: Device IP Address Quick Reference

Device	AS	Device ID	Interface	Interface IP Address
CE1	65001	192.0.2.1/32	ge-0/0/0.0	10.1.1.1/30
PE1	65000	192.0.2.2/32	ge-0/0/2.0	10.1.1.2/30
			ge-0/0/0.0	10.1.2.5/30
			ge-0/0/1.0	10.1.3.9/30
PE2	65000	192.0.2.3/32	ge-0/0/0.0	10.1.2.6/30

Table 11: Device IP Address Quick Reference *(Continued)*

Device	AS	Device ID	Interface	Interface IP Address
			ge-0/0/1.0	10.1.4.13/30
			ge-0/0/2.0	10.1.6.21/30
PE3	65000	192.0.2.4/32	ge-0/0/1.0	10.1.3.10/30
			ge-0/0/0.0	10.1.5.18/30
			ge-0/0/2.0	10.1.7.25/30
			ge-0/0/3.0	10.1.8.29/30
P	65000	192.0.2.5/32	ge-0/0/1.0	10.1.4.14/30
			ge-0/0/0.0	10.1.5.17/30
CE2	65002	192.0.2.6/32	ge-0/0/1.0	10.1.6.22/30
			ge-0/0/2.0	10.1.7.26/30
CE3	65003	192.0.2.7/32	ge-0/0/3.0	10.1.8.30/30

NOTE: This example was tested using logical systems (logical routers). Therefore all the physical interfaces in the example are the same and the configuration is done on separate logical interfaces. In an non-test network, you will use separate physical routers and separate physical interfaces for the connections to other devices.

Configuration

IN THIS SECTION

Procedure

CLI Quick Configuration

To quickly configure this example, copy the following commands, paste them into a text file, remove any line breaks, change any details necessary to match your network configuration, and then copy and paste the commands into the CLI at the `[edit]` hierarchy level.

Device CE1

```
set interfaces ge-0/0/0 unit 0 family inet address 10.1.1.1/30
set interfaces ge-0/0/0 unit 0 family mpls
set interfaces ge-0/0/0 unit 0 description toPE1
set interfaces lo0 unit 0 family inet address 192.0.2.1/32
set routing-options router-id 192.0.2.1
set routing-options autonomous-system 65001
set protocols bgp group toPE1 type external
set protocols bgp group toPE1 export send-direct
set protocols bgp group toPE1 peer-as 65000
set protocols bgp group toPE1 neighbor 10.1.1.2
set policy-options policy-statement send-direct from protocol direct
set policy-options policy-statement send-direct then accept
```

Device PE1

```
set interfaces ge-0/0/2 unit 0 family inet address 10.1.1.2/30
set interfaces ge-0/0/2 unit 0 family mpls
set interfaces ge-0/0/2 unit 0 description toCE1
set interfaces ge-0/0/0 unit 0 family inet address 10.1.2.5/30
set interfaces ge-0/0/0 unit 0 family mpls
set interfaces ge-0/0/0 unit 0 description toPE2
set interfaces ge-0/0/1 unit 0 family inet address 10.1.3.9/30
```

```
set interfaces ge-0/0/1 unit 0 family mpls
set interfaces ge-0/0/1 unit 0 description toPE3
set interfaces lo0 unit 0 family inet address 192.0.2.2/32
set protocols mpls interface all
set protocols ldp interface all
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set protocols ospf area 0.0.0.0 interface ge-0/0/0.0 metric 10
set protocols ospf area 0.0.0.0 interface ge-0/0/1.0 metric 10
set protocols bgp group toInternal type internal
set protocols bgp group toInternal family inet-vpn unicast
set protocols bgp group toInternal local-address 192.0.2.2
set protocols bgp group toInternal neighbor 192.0.2.3
set protocols bgp group toInternal neighbor 192.0.2.4
set routing-options router-id 192.0.2.2
set routing-options autonomous-system 65000
set routing-options forwarding-table export lb
set routing-instances toCE1 instance-type vrf
set routing-instances toCE1 interface ge-0/0/2.0
set routing-instances toCE1 route-distinguisher 65000:1
set routing-instances toCE1 vrf-target target:65000:1
set routing-instances toCE1 vrf-table-label
set routing-instances toCE1 protocols bgp group toCE1 type external
set routing-instances toCE1 protocols bgp group toCE1 peer-as 65001
set routing-instances toCE1 protocols bgp group toCE1 neighbor 10.1.1.1
set routing-instances toCE1 routing-options multipath vpn-unequal-cost equal-external-internal
set policy-options policy-statement lb then load-balance per-packet
```

Device PE2

```
set interfaces ge-0/0/0 unit 0 family inet address 10.1.2.6/30
set interfaces ge-0/0/0 unit 0 family mpls
set interfaces ge-0/0/0 unit 0 description toPE1
set interfaces ge-0/0/1 unit 0 family inet address 10.1.4.13/30
set interfaces ge-0/0/1 unit 0 family mpls
set interfaces ge-0/0/1 unit 0 description toP
set interfaces ge-0/0/2 unit 0 family inet address 10.1.6.21/30
set interfaces ge-0/0/2 unit 0 family mpls
set interfaces ge-0/0/2 unit 0 description toCE2
set interfaces lo0 unit 0 family inet address 192.0.2.3/32
set protocols mpls interface all
set protocols ldp interface all
set protocols ospf area 0.0.0.0 interface lo0.0 passive
```

```
set protocols ospf area 0.0.0.0 interface ge-0/0/0.0 metric 10
set protocols ospf area 0.0.0.0 interface ge-0/0/1.0 metric 5
set protocols bgp group toInternal type internal
set protocols bgp group toInternal family inet-vpn unicast
set protocols bgp group toInternal local-address 192.0.2.3
set protocols bgp group toInternal neighbor 192.0.2.2
set protocols bgp group toInternal neighbor 192.0.2.4
set routing-options router-id 192.0.2.3
set routing-options autonomous-system 65000
set routing-options forwarding-table export lb
set routing-instances toCE2 instance-type vrf
set routing-instances toCE2 interface ge-0/0/2.0
set routing-instances toCE2 route-distinguisher 65000:1
set routing-instances toCE2 vrf-target target:65000:1
set routing-instances toCE2 vrf-table-label
set routing-instances toCE2 protocols bgp group toCE2 type external
set routing-instances toCE2 protocols bgp group toCE2 peer-as 65002
set routing-instances toCE2 protocols bgp group toCE2 neighbor 10.1.6.22
set routing-instances toCE2 routing-options multipath vpn-unequal-cost equal-external-internal
set policy-options policy-statement lb then load-balance per-packet
```

Device PE3

```
set interfaces ge-0/0/1 unit 0 family inet address 10.1.3.10/30
set interfaces ge-0/0/1 unit 0 family mpls
set interfaces ge-0/0/1 unit 0 description toPE1
set interfaces ge-0/0/0 unit 0 family inet address 10.1.5.18/30
set interfaces ge-0/0/0 unit 0 family mpls
set interfaces ge-0/0/0 unit 0 description toP
set interfaces ge-0/0/2 unit 0 family inet address 10.1.7.25/30
set interfaces ge-0/0/2 unit 0 family mpls
set interfaces ge-0/0/2 unit 0 description toCE2
set interfaces ge-0/0/3 unit 0 family inet address 10.1.8.29/30
set interfaces ge-0/0/3 unit 0 family mpls
set interfaces ge-0/0/3 unit 0 description toCE3
set interfaces lo0 unit 0 family inet address 192.0.2.4/32
set protocols mpls interface all
set protocols ldp interface all
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set protocols ospf area 0.0.0.0 interface ge-0/0/1.0 metric 10
set protocols ospf area 0.0.0.0 interface ge-0/0/0.0 metric 5
set protocols bgp group toInternal type internal
```

```
set protocols bgp group toInternal local-address 192.0.2.4
set protocols bgp group toInternal family inet-vpn unicast
set protocols bgp group toInternal family route-target
set protocols bgp group toInternal neighbor 192.0.2.2
set protocols bgp group toInternal neighbor 192.0.2.3
set routing-options router-id 192.0.2.4
set routing-options autonomous-system 65000
set routing-options forwarding-table export lb
set routing-instances toCE2_3 instance-type vrf
set routing-instances toCE2_3 interface ge-0/0/2.0
set routing-instances toCE2_3 interface ge-0/0/3.0
set routing-instances toCE2_3 route-distinguisher 65000:1
set routing-instances toCE2_3 vrf-target target:65000:1
set routing-instances toCE2_3 vrf-table-label
set routing-instances toCE2_3 protocols bgp group toCE2 type external
set routing-instances toCE2_3 protocols bgp group toCE2 peer-as 65002
set routing-instances toCE2_3 protocols bgp group toCE2 neighbor 10.1.7.26
set routing-instances toCE2_3 protocols bgp group toCE3 type external
set routing-instances toCE2_3 protocols bgp group toCE3 peer-as 65003
set routing-instances toCE2_3 protocols bgp group toCE3 neighbor 10.1.8.30
set routing-instances toCE2_3 routing-options multipath vpn-unequal-cost equal-external-internal
set policy-options policy-statement lb then load-balance per-packet
```

Device P

```
set interfaces ge-0/0/1 unit 0 family inet address 10.1.4.14/30
set interfaces ge-0/0/1 unit 0 family mpls
set interfaces ge-0/0/1 unit 0 description toPE2
set interfaces ge-0/0/0 unit 0 family inet address 10.1.5.17/30
set interfaces ge-0/0/0 unit 0 family mpls
set interfaces ge-0/0/0 unit 0 description toPE3
set interfaces lo0 unit 0 family inet address 192.0.2.5/32
set protocols mpls interface all
set protocols ldp interface all
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set protocols ospf area 0.0.0.0 interface ge-0/0/1.0 metric 5
set protocols ospf area 0.0.0.0 interface ge-0/0/0.0 metric 5
set routing-options router-id 192.0.2.5
set routing-options autonomous-system 65000
```

Device CE2

```
set interfaces ge-0/0/1 unit 0 family inet address 10.1.6.22/30
set interfaces ge-0/0/1 unit 0 family mpls
set interfaces ge-0/0/1 unit 0 description toPE2
set interfaces ge-0/0/2 unit 0 family inet address 10.1.7.26/30
set interfaces ge-0/0/2 unit 0 family mpls
set interfaces ge-0/0/2 unit 0 description toPE3
set interfaces lo0 unit 0 family inet address 192.0.2.6/32
set routing-options router-id 192.0.2.6
set routing-options autonomous-system 65002
set protocols bgp group toPE2PE3 type external
set protocols bgp group toPE2PE3 export send-direct
set protocols bgp group toPE2PE3 peer-as 65000
set protocols bgp group toPE2PE3 neighbor 10.1.6.21
set protocols bgp group toPE2PE3 neighbor 10.1.7.25
set policy-options policy-statement send-direct from protocol direct
set policy-options policy-statement send-direct then accept
```

Device CE3

```
set interfaces ge-0/0/3 unit 0 family inet address 10.1.8.30/30
set interfaces ge-0/0/3 unit 0 family mpls
set interfaces ge-0/0/3 unit 0 description toPE3
set interfaces lo0 unit 0 family inet address 192.0.2.7/32
set routing-options router-id 192.0.2.7
set routing-options autonomous-system 65003
set protocols bgp group toPE3 type external
set protocols bgp group toPE3 export send-direct
set protocols bgp group toPE3 peer-as 65000
set protocols bgp group toPE3 neighbor 10.1.8.29
set policy-options policy-statement send-direct from protocol direct
set policy-options policy-statement send-direct then accept
```

Step-by-Step Procedure

The following example requires that you navigate various levels in the configuration hierarchy. For information about navigating the CLI, see *Using the CLI Editor in Configuration Mode* in the CLI User Guide.

To configure unequal-cost load balancing across the VPN setup:

1. Configure the router ID on Device CE1, and assign the device to its autonomous system.

   ```
   [edit routing-options]
   user@CE1# set router-id 192.0.2.1
   user@CE1# set autonomous-system 65001
   ```

   Similarly, configure all other devices.

2. Configure BGP groups for traffic through the entire network.

   a. Configure the BGP group for traffic to and from the MPLS network (CE devices).

      ```
      [edit protocols bgp group toPE1]
      user@CE1# set type external
      user@CE1# set peer-as 65000
      user@CE1# set neighbor 10.1.1.2
      ```

   b. Configure similar BGP groups (**`to AS 65000`** and `toPE3`) on Devices CE2 and CE3 by modifying the `peer-as` and `neighbor` statements accordingly.

   c. Configure the BGP group for traffic through the MPLS network (PE devices).

      ```
      [edit protocols bgp group toInternal]
      user@PE1# set type internal
      user@PE1# set family inet-vpn unicast
      user@PE1# set local-address 192.0.2.2
      user@PE1# set neighbor 192.0.2.3
      user@PE1# set neighbor 192.0.2.4
      ```

   d. Configure the same BGP group (`toInternal`) on Devices PE2 and PE3 by modifying the `local-address` and `neighbor` statements accordingly.

3. Configure a routing policy for exporting routes to and from the MPLS network (`send-direct` policy) and a policy for load balancing traffic network across the MPLS network (`lb` policy).

a. Configure a policy (`send-direct`) for exporting routes from the routing table into BGP on Device CE1.

```
[edit policy-options policy-statement send-direct]
user@CE1# set from protocol direct
user@CE1# set then accept
```

```
[edit protocols bgp group toPE1]
user@CE1# set export send-direct
```

Similarly, configure the `send-direct` policy on Devices CE2 and CE3.

b. Configure a policy (`lb`) for exporting routes from the routing table into the forwarding table on Device PE1.

The `lb` policy configures per-packet load balancing, which ensures that all next-hop addresses for a destination are installed in the forwarding table.

```
[edit policy-options policy-statement lb]
user@PE1# set then load-balance per-packet
```

```
[edit routing-options]
user@PE1# set forwarding-table export lb
```

Similarly, configure the `lb` policy on Devices PE2, and PE3.

4. Configure the following:

   a. Configure the routing instance on the PE devices for exporting routes through the autonomous systems.

   b. Include the `equal-external-internal` statement at the `[edit routing-instances instance-name routing-options multipath vpn-unequal-cost]` hierarchy level to enable load balancing in the network.

   c. Include the `vrf-table-label` statement at the `[edit routing-instances instance-name]` hierarchy level for filtering traffic prior to exiting the egress device (Device CE3).

Device PE1

```
[edit routing-instances toCE1]
user@PE1# set instance-type vrf
user@PE1# set interface ge-0/0/2.0
user@PE1# set route-distinguisher 65000:1
user@PE1# set vrf-target target:65000:1
user@PE1# set vrf-table-label
user@PE1# set protocols bgp group toCE1 type external
user@PE1# set protocols bgp group toCE1 peer-as 65001
user@PE1# set protocols bgp group toCE1 neighbor 10.1.1.1
user@PE1# set routing-options multipath vpn-unequal-cost equal-external-internal
```

Device PE2

```
[edit routing-instances toCE1]
user@PE2# set instance-type vrf
user@PE2# set interface ge-0/0/2.0
user@PE2# set route-distinguisher 65000:1
user@PE2# set vrf-target target:65000:1
user@PE2# set vrf-table-label
user@PE2# set protocols bgp group toCE2 type external
user@PE2# set protocols bgp group toCE2 peer-as 65002
user@PE2# set protocols bgp group toCE2 neighbor 10.1.6.22
user@PE2# set routing-options multipath vpn-unequal-cost equal-external-internal
```

Device PE3

```
[edit routing-instances toCE2_3]
user@PE3# set instance-type vrf
user@PE3# set interface ge-0/0/2.0
user@PE3# set interface ge-0/0/3.0
user@PE3# set route-distinguisher 65000:1
user@PE3# set vrf-target target:65000:1
user@PE3# set vrf-table-label
user@PE3# set protocols bgp group toCE2 type external
user@PE3# set protocols bgp group toCE2 peer-as 65002
user@PE3# set protocols bgp group toCE2 neighbor 10.1.7.26
user@PE3# set protocols bgp group toCE3 type external
user@PE3# set protocols bgp group toCE3 peer-as 65003
```

```
user@PE3# set protocols bgp group toCE3 neighbor 10.1.8.30
user@PE3# set routing-options multipath vpn-unequal-cost equal-external-internal
```

Results

From configuration mode, confirm your configuration by entering the `show configuration` command. If the output does not display the intended configuration, repeat the instructions in this example to correct the configuration. The following is a snippet from the `show configuration` output for PE3.

```
user@PE3# show configuration
interfaces {
    ge-0/0/0 {
        unit 0 {
            description toP;
            family inet {
                address 10.1.5.18/30;
            }
            family mpls;
        }
    }
    ge-0/0/1 {
        unit 0 {
            description toPE1;
            family inet {
                address 10.1.3.10/30;
            }
            family mpls;
        }
    }
    ge-0/0/2 {
        unit 0 {
            description toCE2;
            family inet {
                address 10.1.7.25/30;
            }
            family mpls;
        }
    }
    ge-0/0/3 {
        unit 0 {
            description toCE3;
```

```
            family inet {
                address 10.1.8.29/30;
            }
            family mpls;
        }
    }
    lo0 {
        unit 0 {
            family inet {
                address 192.0.2.4/32;
            }
        }
    }
}
policy-options {
    policy-statement lb {
        then {
            load-balance per-packet;
        }
    }
}
routing-instances {
    toCE2_3 {
        routing-options {
            multipath {
                vpn-unequal-cost equal-external-internal;
            }
        }
        protocols {
            bgp {
                group toCE2 {
                    type external;
                    peer-as 65002;
                    neighbor 10.1.7.26;
                }
                group toCE3 {
                    type external;
                    peer-as 65003;
                    neighbor 10.1.8.30;
                }
            }
        }
        instance-type vrf;
```

```
        interface ge-0/0/2.0;
        interface ge-0/0/3.0;
        route-distinguisher 65000:1;
        vrf-target target:65000:1;
        vrf-table-label;
    }
}
routing-options {
    forwarding-table {
        export lb;
    }
    router-id 192.0.2.4;
    autonomous-system 65000;
}
protocols {
    ospf {
        area 0.0.0.0 {
            interface lo0.0 {
                passive;
            }
            interface ge-0/0/1.0 {
                metric 10;
            }
            interface ge-0/0/0.0 {
                metric 5;
            }
        }
    }
    bgp {
        group toInternal {
            type internal;
            local-address 192.0.2.4;
            family inet-vpn {
                unicast;
            }
            family route-target;
            neighbor 192.0.2.2;
            neighbor 192.0.2.3;
        }
    }
    ldp {
        interface all;
    }
```

```
    mpls {
        interface all;
    }
}
```

If you are done configuring the device, enter `commit` from configuration mode.

Verification

IN THIS SECTION

Confirm that the configuration is working properly.

Verifying BGP

Purpose

Verify that BGP is working.

Action

From operational mode, run the `show route protocol bgp` command.

```
user@PE3> show route protocol bgp

inet.0: 19 destinations, 19 routes (19 active, 0 holddown, 0 hidden)

inet.3: 3 destinations, 3 routes (3 active, 0 holddown, 0 hidden)

toCE2_3.inet.0: 9 destinations, 14 routes (9 active, 0 holddown, 0 hidden)
@ = Routing Use Only, # = Forwarding Use Only
+ = Active Route, - = Last Active, * = Both

10.1.1.0/30        *[BGP/170] 6d 19:24:16, localpref 100, from 192.0.2.2
                      AS path: I, validation-state: unverified
```

```
                    >  to 10.1.3.9 via ge-0/0/1.0, Push 16
10.1.6.20/30       *[BGP/170] 6d 19:24:12, localpref 100, from 192.0.2.3
                      AS path: I, validation-state: unverified
                    >  to 10.1.5.17 via ge-0/0/0.0, Push 16, Push 299808(top)
                    [BGP/170] 6d 17:53:50, localpref 100
                      AS path: 65002 I, validation-state: unverified
                    >  to 10.1.7.26 via ge-0/0/2.0
10.1.7.24/30        [BGP/170] 6d 17:53:50, localpref 100
                      AS path: 65002 I, validation-state: unverified
                    >  to 10.1.7.26 via ge-0/0/2.0
10.1.8.28/30        [BGP/170] 6d 17:55:03, localpref 100
                      AS path: 65003 I, validation-state: unverified
                    >  to 10.1.8.30 via ge-0/0/3.0
192.0.2.1/32       *[BGP/170] 6d 19:24:16, localpref 100, from 192.0.2.2
                      AS path: 65001 I, validation-state: unverified
                    >  to 10.1.3.9 via ge-0/0/1.0, Push 16
192.0.2.6/32       @[BGP/170] 6d 17:53:50, localpref 100
                      AS path: 65002 I, validation-state: unverified
                    >  to 10.1.7.26 via ge-0/0/2.0
                    [BGP/170] 6d 17:53:50, localpref 100, from 192.0.2.3
                      AS path: 65002 I, validation-state: unverified
                    >  to 10.1.5.17 via ge-0/0/0.0, Push 16, Push 299808(top)
192.0.2.7/32       *[BGP/170] 6d 17:55:03, localpref 100
                      AS path: 65003 I, validation-state: unverified
                    >  to 10.1.8.30 via ge-0/0/3.0

mpls.0: 12 destinations, 12 routes (12 active, 0 holddown, 0 hidden)

bgp.l3vpn.0: 4 destinations, 4 routes (4 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

65000:1:10.1.1.0/30
                   *[BGP/170] 6d 19:24:16, localpref 100, from 192.0.2.2
                      AS path: I, validation-state: unverified
                    >  to 10.1.3.9 via ge-0/0/1.0, Push 16
65000:1:10.1.6.20/30
                   *[BGP/170] 6d 19:24:12, localpref 100, from 192.0.2.3
                      AS path: I, validation-state: unverified
                    >  to 10.1.5.17 via ge-0/0/0.0, Push 16, Push 299808(top)
65000:1:192.0.2.1/32
                   *[BGP/170] 6d 19:24:16, localpref 100, from 192.0.2.2
                      AS path: 65001 I, validation-state: unverified
                    >  to 10.1.3.9 via ge-0/0/1.0, Push 16
```

```
65000:1:192.0.2.6/32
                   *[BGP/170] 6d 17:53:50, localpref 100, from 192.0.2.3
                      AS path: 65002 I, validation-state: unverified
                    >  to 10.1.5.17 via ge-0/0/0.0, Push 16, Push 299808(top)

inet6.0: 1 destinations, 1 routes (1 active, 0 holddown, 0 hidden)

toCE2_3.inet6.0: 1 destinations, 1 routes (1 active, 0 holddown, 0 hidden)

bgp.rtarget.0: 2 destinations, 2 routes (2 active, 0 holddown, 0 hidden)
```

The output lists the BGP routes installed into the routing table. The lines of output that start with **192.0.2.1/32**, **10.1.1.0/30**, and **65000:1:192.0.2.1/32** show the BGP routes to Device CE1, which is in AS 65001. The lines of output that start with **192.0.2.6/32**, **65000:1:192.0.2.6/32**, and **65000:1:10.1.6.20/30** show the BGP routes to Device CE2, which is in AS 65002. The line of output that starts with **192.0.2.7/32** shows the BGP route to Device CE3, which is in AS 65003.

Meaning

BGP is functional in the network.

Verifying Load Balancing

Purpose

Verify that forwarding is taking place in both directions by checking:

- If both next hops are installed in the forwarding table for a route.
- If external BGP routes are installed in the forwarding table for a route.

Action

From operational mode, run the **show route forwarding-table** and **show route forwarding-table destination <destination IP>** commands.

```
user@PE3> show route forwarding-table

Routing table: default.inet
Internet:
```

```
Destination        Type RtRef Next hop           Type Index    NhRef Netif
default            perm     1                    rjct       36     2
0.0.0.0/32         perm     0                    dscd       34     1
10.1.2.4/30        user     0                    ulst  1048576     2
                              10.1.5.17          ucst      625    12 ge-0/0/0.0
                              10.1.3.9           ucst      617     8 ge-0/0/1.0
10.1.3.8/30        intf     0                    rslv      604     1 ge-0/0/1.0
10.1.3.8/32        dest     0 10.1.3.8           recv      602     1 ge-0/0/1.0
10.1.3.9/32        dest     0 0:50:56:93:63:e7   ucst      617     8 ge-0/0/1.0
10.1.3.10/32       intf     0 10.1.3.10          locl      603     2
10.1.3.10/32       dest     0 10.1.3.10          locl      603     2
10.1.3.11/32       dest     0 10.1.3.11          bcst      601     1 ge-0/0/1.0
10.1.4.12/30       user     0 10.1.5.17          ucst      625    12 ge-0/0/0.0
10.1.5.16/30       intf     0                    rslv      600     1 ge-0/0/0.0
10.1.5.16/32       dest     0 10.1.5.16          recv      598     1 ge-0/0/0.0
10.1.5.17/32       dest     0 0:50:56:93:f8:ff   ucst      625    12 ge-0/0/0.0
10.1.5.18/32       intf     0 10.1.5.18          locl      599     2
10.1.5.18/32       dest     0 10.1.5.18          locl      599     2
10.1.5.19/32       dest     0 10.1.5.19          bcst      597     1 ge-0/0/0.0
192.0.2.2/32       user     1 10.1.3.9           ucst      617     8 ge-0/0/1.0
192.0.2.3/32       user     1 10.1.5.17          ucst      625    12 ge-0/0/0.0
192.0.2.4/32       intf     0 192.0.2.4          locl      607     1
192.0.2.5/32       user     1 10.1.5.17          ucst      625    12 ge-0/0/0.0
224.0.0.0/4        perm     1                    mdsc       35     1
224.0.0.1/32       perm     0 224.0.0.1          mcst       31     5
224.0.0.2/32       user     1 224.0.0.2          mcst       31     5
224.0.0.5/32       user     1 224.0.0.5          mcst       31     5
255.255.255.255/32 perm     0                    bcst       32     1

Routing table: __master.anon__.inet
Internet:
Destination        Type RtRef Next hop           Type Index    NhRef Netif
default            perm     0                    rjct      524     1
0.0.0.0/32         perm     0                    dscd      522     1
224.0.0.0/4        perm     0                    mdsc      523     1
224.0.0.1/32       perm     0 224.0.0.1          mcst      526     1
255.255.255.255/32 perm     0                    bcst      527     1

Routing table: toCE2_3.inet
Internet:
Destination        Type RtRef Next hop           Type Index    NhRef Netif
default            perm     0                    rjct      566     1
0.0.0.0/32         perm     0                    dscd      564     1
```

```
10.1.1.0/30        user     0                   indr  1048574     3
                              10.1.3.9          Push 16      623     2 ge-0/0/1.0
10.1.6.20/30       user     0                   indr  1048575     3
                              10.1.5.17         Push 16, Push 299808(top)      629     2
ge-0/0/0.0
10.1.7.24/30       intf     0                   rslv      616     1 ge-0/0/2.0
10.1.7.24/32       dest     0 10.1.7.24         recv      614     1 ge-0/0/2.0
10.1.7.25/32       intf     0 10.1.7.25         locl      615     2
10.1.7.25/32       dest     0 10.1.7.25         locl      615     2
10.1.7.26/32       dest     1 0:50:56:93:86:18   ucst      618     4 ge-0/0/2.0
10.1.7.27/32       dest     0 10.1.7.27         bcst      613     1 ge-0/0/2.0
10.1.8.28/30       intf     0                   rslv      612     1 ge-0/0/3.0
10.1.8.28/32       dest     0 10.1.8.28         recv      610     1 ge-0/0/3.0
10.1.8.29/32       intf     0 10.1.8.29         locl      611     2
10.1.8.29/32       dest     0 10.1.8.29         locl      611     2
10.1.8.30/32       dest     1 0:50:56:93:bb:ed   ucst      622     4 ge-0/0/3.0
10.1.8.31/32       dest     0 10.1.8.31         bcst      609     1 ge-0/0/3.0
192.0.2.1/32       user     0                   indr  1048574     3
                              10.1.3.9          Push 16      623     2 ge-0/0/1.0
192.0.2.6/32       user     0                   ulst  1048577     2
                              10.1.7.26         ucst      618     4 ge-0/0/2.0
                                                indr  1048575     3
                              10.1.5.17         Push 16, Push 299808(top)      629     2
ge-0/0/0.0
192.0.2.7/32       user     0 10.1.8.30         ucst      622     4 ge-0/0/3.0
224.0.0.0/4        perm     0                   mdsc      565     1
224.0.0.1/32       perm     0 224.0.0.1         mcst      568     1
255.255.255.255/32 perm     0                   bcst      569     1
...
```

In the **default.inet** routing table, which is the forwarding table, the line of output that starts with **10.1.2.4/30** shows that for a route to Device PE2 in the same AS, two next hops are installed in the table: **10.1.3.9** and **10.1.5.17**.

In the **toCE2_3.inet** routing table, which is the external routing table, the line of output that starts with **192.0.2.6/32** shows that for a route to Device CE2 in AS 65002, an internal next hop of **10.1.5.17** and an external next hop of **10.1.7.26** are installed in the table. This indicates that both internal and external BGP routes are operational in the network.

```
user@PE3> show route forwarding-table destination 10.1.2.6

Routing table: default.inet
```

```
Internet:
Destination        Type RtRef Next hop           Type Index    NhRef Netif
10.1.2.4/30        user     0                    ulst  1048576     2
                             10.1.5.17           ucst      625    12 ge-0/0/0.0
                             10.1.3.9            ucst      617     8 ge-0/0/1.0

Routing table: __pfe_private__.inet
Internet:
Destination        Type RtRef Next hop           Type Index    NhRef Netif
default            perm     0                    dscd      513     2

Routing table: __master.anon__.inet
Internet:
Destination        Type RtRef Next hop           Type Index    NhRef Netif
default            perm     0                    rjct      524     1

Routing table: __juniper_services__.inet
Internet:
Destination        Type RtRef Next hop           Type Index    NhRef Netif
default            perm     0                    dscd      546     2

Routing table: toCE2_3.inet
Internet:
Destination        Type RtRef Next hop           Type Index    NhRef Netif
default            perm     0                    rjct      630     1
```

The line of output that starts with **10.1.2.4/30** shows that for a route from Device PE3 to Device PE2 in the same AS, two next hops are installed in the table: **10.1.3.9** through the **ge-0/0/1.0** interface, and **10.1.5.17** through the **ge-2/1/10.18** interface.

Meaning

Multiple next hops for a route, including external BGP routes, are installed in the forwarding tables.

SEE ALSO

Example: Load Balancing BGP Traffic

Configuring Protocol-Independent Load Balancing in Layer 3 VPNs

IN THIS SECTION

Protocol-independent load balancing for Layer 3 VPNs allows the forwarding next hops of both the active route and alternative paths to be used for load balancing. Protocol-independent load balancing works in conjunction with Layer 3 VPNs. It supports the load balancing of VPN routes independently of the assigned route distinguisher. When protocol-independent load balancing is enabled, both routes to other PE routers and routes to directly connected CE routers are load-balanced.

When load-balancing information is created for a given route, the active path is marked as `Routing Use Only` in the output of the `show route table` command.

The following sections describe how to configure protocol-independent load balancing and how this configuration can affect routing policies:

Configuring Load Balancing for Layer 3 VPNs

The configuration of protocol-independent load balancing for Layer 3 VPNs is a little different for IPv4 versus IPv6:

- IPv4—You only need to configure the `multipath` statement at either the `[edit routing-instances` *`routing-instance-name`* `routing-options]` hierarchy level or the `[edit routing-instances` *`routing-instance-name`* `routing-options rib` *`routing-table-name`*`]` hierarchy level.
- IPv6—You need to configure the `multipath` statement at both the `[edit routing-instances` *`routing-instance-name`* `routing-options]` hierarchy level and the `[edit routing-instances` *`routing-instance-name`* `routing-options rib` *`routing-table-name`*`]` hierarchy level.

NOTE: You cannot configure the `multipath` statement and sub-statements at the same time that you have configured the *l3vpn* statement.

To configure protocol-independent load balancing for Layer 3 VPNs, include the `multipath` statement:

```
multipath {
    vpn-unequal-cost equal-external-internal;
}
```

When you include the `multipath` statement at the following hierarchy levels, protocol-independent load balancing is applied to the default routing table for that routing instance (*routing-instance-name*.inet.0):

- `[edit routing-instances` *`routing-instance-name`* `routing-options]`
- `[edit logical-systems` *`logical-system-name`* `routing-instances` *`routing-instance-name`* `routing-options]`

NOTE: The `[edit logical-systems]` hierarchy level is not applicable in ACX Series routers.

When you include the `multipath` statement at the following hierarchy levels, protocol-independent load balancing is applied to the specified routing table:

- `[edit routing-instances` *`routing-instance-name`* `routing-options rib` *`routing-table-name`*`]`
- `[edit logical-systems` *`logical-system-name`* `routing-instances` *`routing-instance-name`* `routing-options rib` *`routing-table-name`*`]`

NOTE: The `[edit logical-systems]` hierarchy level is not applicable in ACX Series routers.

The `vpn-unequal-cost` statement is optional:

- When you include it, protocol-independent load balancing is applied to VPN routes that are equal until the IGP metric with regard to route selection.
- When you do not include it, protocol-independent load balancing is applied to VPN routes that are equal until the router identifier with regard to route selection.

NOTE: The `vpn-unequal-cost` statement is not applicable in ACX Series routers.

The `equal-external-internal` statement is also optional. When you include it, protocol-independent load balancing is applied to both internal and external BGP paths. You can configure this in conjunction with egress IP header filtering (enabled with the `vrf-table-label` statement). For more information, see "Load Balancing and IP Header Filtering for Layer 3 VPNs" on page 1151.

NOTE: You can include the `vpn-unequal-cost equal-external-internal` statement and the `l3vpn` statement at the `[edit routing-options forwarding-options chained-composite-next-hop ingress]` hierarchy level simultaneously. However, if you do this, EBGP does not work. This means that when there are both paths with chained next hops and paths with nonchained next hops as candidates for EBGP equal-cost multipath (ECMP), the paths using chained next hops are excluded. In a typical case, the excluded paths are the internal paths.

Configuring Load Balancing and Routing Policies

If you enable protocol-independent load balancing for Layer 3 VPNs by including the `multipath` statement and if you also include the `load-balance per-packet` statement in the routing policy configuration, packets are not load-balanced.

For example, a PE router has the following VRF routing instance configured:

```
[edit routing-instances]
load-balance-example {
    instance-type vrf;
    interface fe-0/1/1.0;
    interface fe-0/1/1.1;
    route-distinguisher 2222:2;
    vrf-target target:2222:2;
    routing-options {
        multipath;
    }
    protocols {
        bgp {
            group group-example {
                import import-policy;
                family inet {
                    unicast;
                }
                export export-policy;
                peer-as 4444;
                local-as 3333;
                multipath;
                as-override;
                neighbor 10.12.33.22;
            }
        }
```

```
    }
}
```

The PE router also has the following policy statement configured:

```
[edit policy-options policy-statement export-policy]
from protocol bgp;
then {
    load-balance per-packet;
}
```

When you include the `multipath` statement in the VRF routing instance configuration, the paths are no longer marked as BGP paths but are instead marked as multipath paths. Packets from the PE router are not load-balanced.

To ensure that VPN load-balancing functions as expected, do not include the `from protocol` statement in the policy statement configuration. The policy statement should be configured as follows:

```
[edit policy-options policy-statement export-policy]
then {
    load-balance per-packet;
}
```

For more information about how to configure per-packet load balancing, see the Routing Policies, Firewall Filters, and Traffic Policers User Guide.

Example: Configuring PIM Join Load Balancing on Next-Generation Multicast VPN

IN THIS SECTION

This example shows how to configure multipath routing for external and internal virtual private network (VPN) routes with unequal interior gateway protocol (IGP) metrics and Protocol Independent Multicast (PIM) join load balancing on provider edge (PE) routers running next-generation multicast VPN (MVPN).

This feature allows customer PIM (C-PIM) join messages to be load-balanced across available internal BGP (IBGP) upstream paths when there is no external BGP (EBGP) path present, and across available EBGP upstream paths when external and internal BGP (EIBGP) paths are present toward the source or rendezvous point (RP).

Requirements

This example uses the following hardware and software components:

- Three routers that can be a combination of M Series, MX Series, or T Series routers.
- Junos OS Release 12.1 running on all the devices.

Before you begin:

1. Configure the device interfaces.
2. Configure the following routing protocols on all PE routers:
   - OSPF
   - MPLS
   - LDP
   - PIM
   - BGP
3. Configure a multicast VPN.

Overview and Topology

Junos OS Release 12.1 and later support multipath configuration along with PIM join load balancing. This allows C-PIM join messages to be load-balanced across all available IBGP paths when there are only IBGP paths present, and across all available upstream EBGP paths when EIBGP paths are present toward the source (or RP). Unlike Draft-Rosen MVPN, next-generation MVPN does not utilize unequal EIBGP paths to send C-PIM join messages. This feature is applicable to IPv4 C-PIM join messages.

By default, only one active IBGP path is used to send the C-PIM join messages for a PE router having only IBGP paths toward the source (or RP). When there are EIBGP upstream paths present, only one active EBGP path is used to send the join messages.

In a next-generation MVPN, C-PIM join messages are translated into (or encoded as) BGP customer multicast (C-multicast) MVPN routes and advertised with the BGP MCAST-VPN address family toward the sender PE routers. A PE router originates a C-multicast MVPN route in response to receiving a C-

PIM join message through its PE router to customer edge (CE) router interface. The two types of C-multicast MVPN routes are:

- Shared tree join route (C-*, C-G)
    - Originated by receiver PE routers.
    - Originated when a PE router receives a shared tree C-PIM join message through its PE-CE router interface.
- Source tree join route (C-S, C-G)
    - Originated by receiver PE routers.
    - Originated when a PE router receives a source tree C-PIM join message (C-S, C-G), or originated by the PE router that already has a shared tree join route and receives a source active autodiscovery route.

    The upstream path in a next-generation MVPN is selected using the Bytewise-XOR hash algorithm as specified in Internet draft draft-ietf-l3vpn-2547bis-mcast, *Multicast in MPLS/BGP IP VPNs*. The hash algorithm is performed as follows:

    1. The PE routers in the candidate set are numbered from lower to higher IP address, starting from 0.
    2. A bytewise exclusive-or of all the bytes is performed on the C-root (source) and the C-G (group) address.
    3. The result is taken modulo *n*, where *n* is the number of PE routers in the candidate set. The result is N.
    4. N represents the IP address of the upstream PE router as numbered in Step 1.

During load balancing, if a PE router with one or more upstream IBGP paths toward the source (or RP) discovers a new IBGP path toward the same source (or RP), the C-PIM join messages distributed among previously existing IBGP paths get redistributed due to the change in the candidate PE router set.

In this example, PE1, PE2, and PE3 are the PE routers that have the multipath PIM join load-balancing feature configured. Router PE1 has two EBGP paths and one IBGP upstream path, PE2 has one EBGP path and one IBGP upstream path, and PE3 has two IBGP upstream paths toward the Source. Router CE4 is the customer edge (CE) router attached to PE3. Source and Receiver are the Free BSD hosts.

On PE routers that have EIBGP paths toward the source (or RP), such as PE1 and PE2, PIM join load balancing is performed as follows:

1. The C-PIM join messages are sent using EBGP paths only. IBGP paths are not used to propagate the join messages.

In Figure 91 on page 1181, the PE1 router distributes the join messages between the two EBGP paths to the CE1 router, and PE2 uses the EBGP path to CE1 to send the join messages.

2. If a PE router loses one or more EBGP paths toward the source (or RP), the RPF neighbor on the multicast tunnel interface is selected based on a hash mechanism.

   On discovering the first EBGP path, only new join messages get load-balanced across available EBGP paths, whereas the existing join messages on the multicast tunnel interface are not redistributed.

   If the EBGP path from the PE2 router to the CE1 router goes down, PE2 sends the join messages to PE1 using the IBGP path. When the EBGP path to CE1 is restored, only new join messages that arrive on PE2 use the restored EBGP path, whereas join messages already sent on the IBGP path are not redistributed.

On PE routers that have only IBGP paths toward the source (or RP), such as the PE3 router, PIM join load balancing is performed as follows:

1. The C-PIM join messages from CE routers get load-balanced only as BGP C-multicast data messages among IBGP paths.

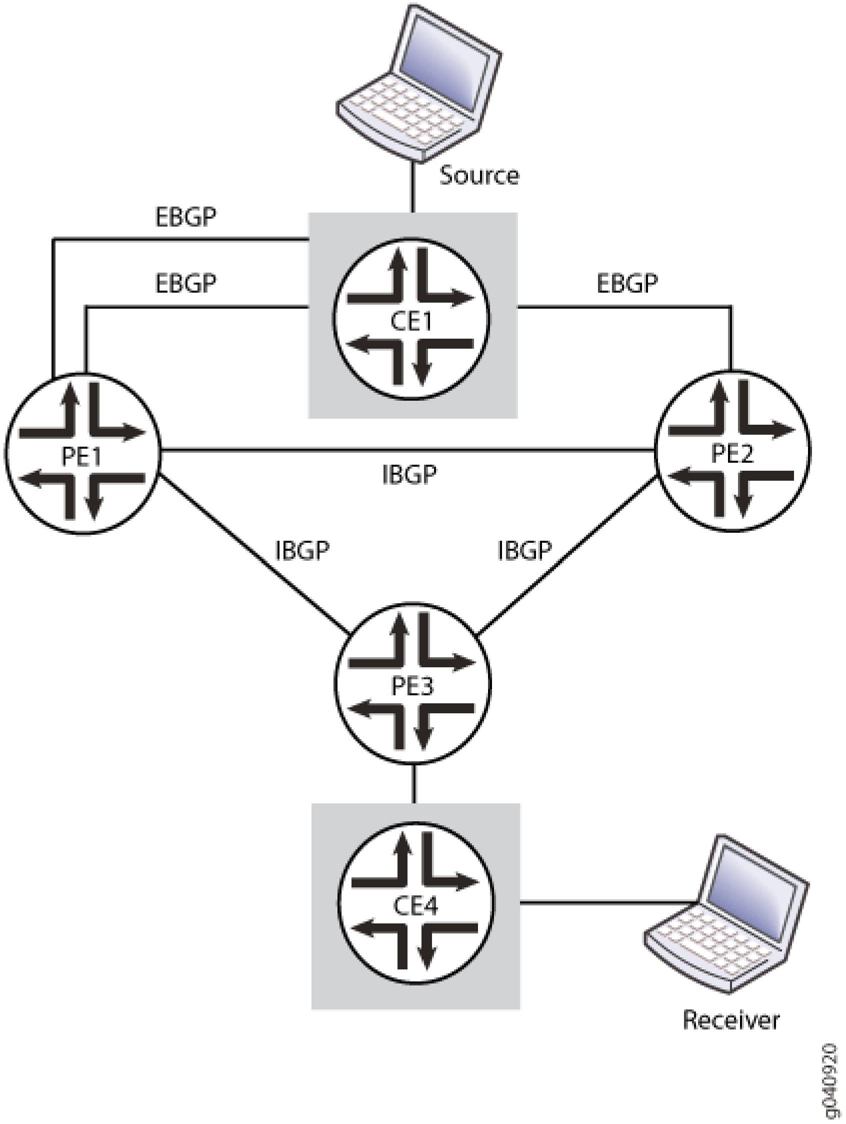
   In Figure 91 on page 1181, assuming that the CE4 host is interested in receiving traffic from the Source, and CE4 initiates source join messages for different groups (Group 1 [C-S,C-G1] and Group 2 [C-S,C-G2]), the source join messages arrive on the PE3 router.

   Router PE3 then uses the Bytewise-XOR hash algorithm to select the upstream PE router to send the C-multicast data for each group. The algorithm first numbers the upstream PE routers from lower to higher IP address starting from **0**.

   Assuming that Router PE1 router is numbered **0** and Router PE2 is **1**, and the hash result for Group 1 and Group 2 join messages is **0** and **1**, respectively, the PE3 router selects PE1 as the upstream PE router to send Group 1 join messages, and PE2 as the upstream PE router to send the Group 2 join messages to the Source.

2. The shared join messages for different groups [C-*,C-G] are also treated in a similar way to reach the destination.

Figure 91: PIM Join Load Balancing on Next-Generation MVPN



Configuration

IN THIS SECTION

CLI Quick Configuration

To quickly configure this example, copy the following commands, paste them into a text file, remove any line breaks, change any details necessary to match your network configuration, copy and paste the commands into the CLI at the `[edit]` hierarchy level, and then enter `commit` from configuration mode.

PE1

```
set routing-instances vpn1 instance-type vrf
set routing-instances vpn1 interface ge-3/0/1.0
set routing-instances vpn1 interface ge-3/3/2.0
set routing-instances vpn1 interface lo0.1
set routing-instances vpn1 route-distinguisher 1:1
set routing-instances vpn1 provider-tunnel rsvp-te label-switched-path-template default-template
set routing-instances vpn1 vrf-target target:1:1
set routing-instances vpn1 vrf-table-label
set routing-instances vpn1 routing-options multipath vpn-unequal-cost equal-external-internal
set routing-instances vpn1 protocols bgp export direct
set routing-instances vpn1 protocols bgp group bgp type external
set routing-instances vpn1 protocols bgp group bgp local-address 10.40.10.1
set routing-instances vpn1 protocols bgp group bgp family inet unicast
set routing-instances vpn1 protocols bgp group bgp neighbor 10.40.10.2 peer-as 3
set routing-instances vpn1 protocols bgp group bgp1 type external
set routing-instances vpn1 protocols bgp group bgp1 local-address 10.10.10.1
set routing-instances vpn1 protocols bgp group bgp1 family inet unicast
set routing-instances vpn1 protocols bgp group bgp1 neighbor 10.10.10.2 peer-as 3
set routing-instances vpn1 protocols pim rp static address 10.255.10.119
set routing-instances vpn1 protocols pim interface all
set routing-instances vpn1  protocols pim join-load-balance
set routing-instances vpn1 protocols mvpn mvpn-mode rpt-spt
set routing-instances vpn1 protocols mvpn mvpn-join-load-balance bytewise-xor-hash
```

PE2

```
set routing-instances vpn1 instance-type vrf
set routing-instances vpn1 interface ge-1/0/9.0
set routing-instances vpn1 interface lo0.1
set routing-instances vpn1 route-distinguisher 2:2
set routing-instances vpn1 provider-tunnel rsvp-te label-switched-path-template default-template
set routing-instances vpn1 vrf-target target:1:1
set routing-instances vpn1 vrf-table-label
set routing-instances vpn1 routing-options multipath vpn-unequal-cost equal-external-internal
```

```
set routing-instances vpn1 protocols bgp export direct
set routing-instances vpn1 protocols bgp group bgp local-address 10.50.10.2
set routing-instances vpn1 protocols bgp group bgp family inet unicast
set routing-instances vpn1 protocols bgp group bgp neighbor 10.50.10.1 peer-as 3
set routing-instances vpn1 protocols pim rp static address 10.255.10.119
set routing-instances vpn1 protocols pim interface all
set routing-instances vpn1 protocols mvpn mvpn-mode rpt-spt
set routing-instances vpn1 protocols mvpn mvpn-join-load-balance bytewise-xor-hash
```

PE3

```
set routing-instances vpn1 instance-type vrf
set routing-instances vpn1 interface ge-0/0/8.0
set routing-instances vpn1 interface lo0.1
set routing-instances vpn1 route-distinguisher 3:3
set routing-instances vpn1 provider-tunnel rsvp-te label-switched-path-template default-template
set routing-instances vpn1 vrf-target target:1:1
set routing-instances vpn1 vrf-table-label
set routing-instances vpn1 routing-options multipath vpn-unequal-cost equal-external-internal
set routing-instances vpn1 routing-options autonomous-system 1
set routing-instances vpn1 protocols bgp export direct
set routing-instances vpn1 protocols bgp group bgp type external
set routing-instances vpn1 protocols bgp group bgp local-address 10.80.10.1
set routing-instances vpn1 protocols bgp group bgp family inet unicast
set routing-instances vpn1 protocols bgp group bgp neighbor 10.80.10.2 peer-as 2
set routing-instances vpn1 protocols pim rp static address 10.255.10.119
set routing-instances vpn1 protocols pim interface all
set routing-instances vpn1 protocols mvpn mvpn-mode rpt-spt
set routing-instances vpn1 protocols mvpn mvpn-join-load-balance bytewise-xor-hash
```

Procedure

Step-by-Step Procedure

The following example requires you to navigate various levels in the configuration hierarchy. For information about navigating the CLI, see *Using the CLI Editor in Configuration Mode*. To configure the PE1 router:

NOTE: Repeat this procedure for every Juniper Networks router in the MVPN domain, after modifying the appropriate interface names, addresses, and any other parameters for each router.

1. Configure a VPN routing forwarding (VRF) routing instance.

```
[edit routing-instances vpn1]
user@PE1# set instance-type vrf
user@PE1# set interface ge-3/0/1.0
user@PE1# set interface ge-3/3/2.0
user@PE1# set interface lo0.1
user@PE1# set route-distinguisher 1:1
user@PE1# set provider-tunnel rsvp-te label-switched-path-template default-template
user@PE1# set vrf-target target:1:1
user@PE1# set vrf-table-label
```

2. Enable protocol-independent load balancing for the VRF instance.

```
[edit routing-instances vpn1]
user@PE1# set routing-options multipath vpn-unequal-cost equal-external-internal
```

3. Configure BGP groups and neighbors to enable PE to CE routing.

```
[edit routing-instances vpn1 protocols]
user@PE1# set bgp export direct
user@PE1# set bgp group bgp type external
user@PE1# set bgp group bgp local-address 10.40.10.1
user@PE1# set bgp group bgp family inet unicast
user@PE1# set bgp group bgp neighbor 10.40.10.2 peer-as 3
user@PE1# set bgp group bgp1 type external
user@PE1# set bgp group bgp1 local-address 10.10.10.1
user@PE1# set bgp group bgp1 family inet unicast
user@PE1# set bgp group bgp1 neighbor 10.10.10.2 peer-as 3
```

4. Configure PIM to enable PE to CE multicast routing.

```
[edit routing-instances vpn1 protocols]
user@PE1# set pim rp static address 10.255.10.119
```

5. Enable PIM on all network interfaces.

```
[edit routing-instances vpn1 protocols]
user@PE1# set pim interface all
```

6. Enable PIM join load balancing for the VRF instance.

```
[edit routing-instances vpn1 protocols]
user@PE1# set pim join-load-balance
```

7. Configure the mode for C-PIM join messages to use rendezvous-point trees, and switch to the shortest-path tree after the source is known.

```
[edit routing-instances vpn1 protocols]
user@PE1# set mvpn mvpn-mode rpt-spt
```

8. Configure the VRF instance to use the Bytewise-XOR hash algorithm.

```
[edit routing-instances vpn1 protocols]
user@PE1# set mvpn mvpn-join-load-balance bytewise-xor-hash
```

Results

From configuration mode, confirm your configuration by entering the **show routing-instances** command. If the output does not display the intended configuration, repeat the instructions in this example to correct the configuration.

```
user@PE1# show routing-instances
routing-instances {
    vpn1 {
        instance-type vrf;
        interface ge-3/0/1.0;
```

```
        interface ge-3/3/2.0;
        interface lo0.1;
        route-distinguisher 1:1;
        provider-tunnel {
            rsvp-te {
                label-switched-path-template {
                    default-template;
                }
            }
        }
        vrf-target target:1:1;
        vrf-table-label;
        routing-options {
            multipath {
                vpn-unequal-cost equal-external-internal;
            }
        }
        protocols {
            bgp {
                export direct;
                group bgp {
                    type external;
                    local-address 10.40.10.1;
                    family inet {
                        unicast;
                    }
                    neighbor 10.40.10.2 {
                        peer-as 3;
                    }
                }
                group bgp1 {
                    type external;
                    local-address 10.10.10.1;
                    family inet {
                        unicast;
                    }
                    neighbor 10.10.10.2 {
                        peer-as 3;
                    }
                }
            }
            pim {
                rp {
```

```
                    static {
                        address 10.255.10.119;
                    }
                }
                interface all;
                join-load-balance;
            }
            mvpn {
                mvpn-mode {
                    rpt-spt;
                }
                mvpn-join-load-balance {
                    bytewise-xor-hash;
                }
            }
        }
    }
```

If you are done configuring the device, enter **commit** from configuration mode.

Verification

IN THIS SECTION

Confirm that the configuration is working properly.

Verifying MVPN C-Multicast Route Information for Different Groups of Join Messages

Purpose

Verify MVPN C-multicast route information for different groups of join messages received on the PE3 router.

Action

From operational mode, run the **show mvpn c-multicast** command.

```
user@PE3>
MVPN instance:
Legend for provider tunnel
I-P-tnl -- inclusive provider tunnel S-P-tnl -- selective provider tunnel
Legend for c-multicast routes properties (Pr)
DS -- derived from (*, c-g)          RM -- remote VPN route
Family : INET

Instance : vpn1
  MVPN Mode : RPT-SPT
  C-mcast IPv4 (S:G)            Ptnl                           St
  0.0.0.0/0:203.0.113.1/24        RSVP-TE P2MP:10.255.10.2, 5834,10.255.10.2
  192.0.2.2/24:203.0.113.1/24       RSVP-TE P2MP:10.255.10.2, 5834,10.255.10.2
  0.0.0.0/0:203.0.113.2/24        RSVP-TE P2MP:10.255.10.14, 47575,10.255.10.14
  192.0.2.2/24:203.0.113.2/24       RSVP-TE P2MP:10.255.10.14, 47575,10.255.10.14
```

Meaning

The output shows how the PE3 router has load-balanced the C-multicast data for the different groups.

- For source join messages (S,G):
  - 192.0.2.2/24:203.0.113.1/24 (S,G1) toward the PE1 router (10.255.10.2 is the loopback address of Router PE1).
  - 192.0.2.2/24:203.0.113.2/24 (S,G2) toward the PE2 router (10.255.10.14 is the loopback address of Router PE2).
- For shared join messages (*,G):
  - 0.0.0.0/0:203.0.113.1/24 (*,G1) toward the PE1 router (10.255.10.2 is the loopback address of Router PE1).
  - 0.0.0.0/0:203.0.113.2/24 (*,G2) toward the PE2 router (10.255.10.14 is the loopback address of Router PE2).

SEE ALSO

PIM Join Load Balancing on Multipath MVPN Routes Overview

Improving Layer 3 VPN Performance

IN THIS SECTION

This topic introduces chained composite next hops (CNHs) and provides an example of how to enable chained CNH on back-to-back PE routers.

Chained Composite Next Hops for VPNs and Layer 2 Circuits

IN THIS SECTION

The Juniper Networks PTX Series Packet Transport Routers, MX Series 5G Universal Routing Platforms with MIC and MPC interfaces, and T4000 Core Routers are principally designed to handle large volumes of traffic in the core of large networks. Chained CNHs help to facilitate this capability by allowing the router to process much larger volumes of routes. A chained CNH allows the router to direct sets of routes sharing the same destination to a common forwarding next hop, rather than having each route also include the destination. In the event that a network destination is changed, rather than having to update all of the routes sharing that destination with the new information, only the shared forwarding next hop is updated with the new information. The chained CNHs continue to point to this forwarding next hop, which now contains the new destination.

When the next hops for MPLS LSPs are created on the routers, the tag information corresponding to the innermost MPLS label is extracted into a chained CNH. The chained CNH is stored in the ingress Packet Forwarding Engine. The chained CNH points to a next hop called the forwarding next hop that resides on the egress Packet Forwarding Engine. The forwarding next hop contains all the other information (all of the labels except for the inner-most labels as well as the IFA/IP information corresponding to the actual next-hop node). Many chained composite next hops can share the same forwarding next hop. Additionally, separating the inner-most label (that is the VPN label) from the forwarding next hop and storing it on the ingress PFE (within the chained composite next hop) helps to conserve egress Packet Forwarding Engine memory by reducing the number of rewrite strings stored on the egress Packet Forwarding Engine.

Table 12 on page 1190 shows support for chained CNHs for ingress or transit routers on the MPLS network.

Table 12: Support for Chained Composite Next Hops

Platform	L2 VPN	L3 VPN	L2 CKT
PTX Series	Ingress and transit	Ingress and transit	Ingress only
MX Series	Ingress only	Ingress only	Ingress only

To enable chained CNHs on a T4000 router, the chassis must be configured to use the `enhanced-mode` option in network services mode.

Benefits of chained composite next hops

Chained CNH optimizes the memory and performance of the router by reducing the size of the forwarding table. The router can use the same next-hop entry in the forwarding table for routes with different destinations when the next-hop is the same. This reduces the number of entries in the forwarding table and reduces the number of changes when the next hop entry has to be modified.

Accepting Route Updates with Unique Inner VPN Labels in Layer 3 VPNs

IN THIS SECTION

For Layer 3 VPNs configured on Juniper Networks routers, Junos OS normally allocates one inner VPN label for each customer edge (CE)-facing virtual routing and forwarding (VRF) interface of a provider edge (PE) router. However, other vendors allocate one VPN label for each route learned over the CE-facing interfaces of a PE router. This practice increases the number of VPN labels exponentially, which leads to slow system processing and slow convergence time.

Chained CNHs is a composition function that concatenates the partial rewrite strings associated with individual next hops to form a larger rewrite string that is added to a packet. By using this function, the number of routes with unique inner VPN labels that can be processed by a Juniper Networks router is increased substantially. Common route update elements associated with Layer 3 VPNs are combined, reducing the number of route updates and individual states the Juniper Networks router must maintain, and leading to enhanced scaling and convergence performance.

NOTE: ACX Series routers supports the `chained-composite-next-hop ingress` CLI statement at the `[edit routing-options forwarding-table]` hierarchy level only for Layer 3 VPNs. The `chained-composite-next-hop ingress` CLI statement for Layer 2 services is not supported.

You can configure the router based on the number of VPN labels you want to manage and on whether or not you want to create chained CNHs for IPv6-labeled routes:

Accepting Up to One Million Layer 3 VPN Route Updates

For Juniper Networks routers participating in a mixed vendor network with up to one million Layer 3 VPN labels, include the `l3vpn` statement at the `[edit routing-options forwarding-table chained-composite-next-hop ingress]` hierarchy level. The `l3vpn` statement is disabled by default.

NOTE: ACX Series routers do not support the `chained-composite-next-hop ingress` CLI statement at the `[edit routing-options forwarding-table] hierarchy level.`

BEST PRACTICE: We recommend that you configure the `l3vpn` statement whenever you have deployed Juniper Networks routers in mixed vendor networks of up to one million routes to support Layer 3 VPNs.

Because using this statement can also enhance the Layer 3 VPN performance of Juniper Networks routers in networks where only Juniper Networks routers are deployed, we recommend configuring the statements in these networks as well.

You can configure the `l3vpn` statement on the following routers:

- ACX Series routers
- MX Series routers
- M120 routers

- M320 routers with one or more Enhanced III FPCs
- T Series routers (for Junos OS Release 10.4 and later)

To accept up to one million Layer 3 VPN route updates with unique inner VPN labels, configure the `l3vpn` statement. This statement is supported on indirectly connected PE routers only. Configuring this statement on a router that is directly connected to a PE router provides no benefit. You can configure the `l3vpn` statement on a router with a mix of links to both directly connected and indirectly connected PE routers.

NOTE: You cannot configure the `l3vpn` statement and sub-statements at same time that you have configured the *vpn-unequal-cost* statement.

To configure the router to accept up to one million Layer 3 VPN route updates with unique inner VPN labels:

1. Include the `l3vpn` statement.

   ```
   [edit routing-options forwarding-table chained-composite-next-hop ingress]
   user@host>set l3vpn
   ```

2. To enhance memory allocation to support a larger number of Layer 3 VPN labels, include the `vpn-label` statement.

   ```
   [edit chassis memory-enhanced]
   user@host>set vpn-label
   ```

   NOTE: The `vpn-label` statement does not provide any functional changes when used on the MX Series routers. You can omit the configuration of this statement on MX Series routers.

   For more information about configuring more memory for Layer 3 VPN labels, see the Junos OS Administration Library.

After you have configured the `l3vpn` statement, you can determine whether or not a Layer 3 VPN route is a part of a chained CNH by examining the display output of the following commands:

- `show route` *`route-value`* `extensive`
- `show route forwarding-table destination` *`destination-value`* `extensive`

Accepting More Than One Million Layer 3 VPN Route Updates

For Juniper Networks routers participating in a mixed vendor network with more than one million Layer 3 VPN labels, include the `extended-space` statement at the `[edit routing-options forwarding-table chained-composite-next-hop ingress l3vpn]` hierarchy level. The `extended-space` statement is disabled by default.

> **NOTE**: The `chained-composite-next-hop ingress` and `extended-space` statements are not supported on ACX Series routers.

> **BEST PRACTICE**: We recommend that you configure the `extended-space` statement in mixed vendor networks containing more than one million routes to support Layer 3 VPNs.
>
> Because using this statements can also enhance the Layer 3 VPN performance of Juniper Networks routers in networks where only Juniper Networks routers are deployed, we recommend configuring the statement in these networks as well.

Using the `extended-space` statement can double the number of routes with unique inner VPN labels that can be processed by a Juniper Networks router. However, when configuring such very large-scale Layer 3 VPN scenarios, keep the following guidelines in mind:

- The `extended-space` statement is supported only on MX Series routers containing only MPCs.
- The chassis must be configured to use the `enhanced-ip` option in network services mode.

  For more information about configuring chassis network services, see the Junos OS Administration Library.
- Ensure that you configure per-packet load balancing for associated policies.

  For more information about configuring policies, see the Routing Policies, Firewall Filters, and Traffic Policers User Guide.

> **BEST PRACTICE**: We strongly recommend using 64-bit routing engines running 64-bit Junos OS to support Layer 3 VPN prefixes with unique inner VPN labels at higher scale.

To configure the router to accept more than one million Layer 3 VPN route updates with unique inner VPN labels:

1. Include the `l3vpn` statement.

```
[edit routing-options forwarding-table chained-composite-next-hop ingress]
user@host>set l3vpn
```

2. Include the `extended-space` statement.

```
[edit routing-options forwarding-table chained-composite-next-hop ingress l3vpn]
user@host> set extended-space
```

3. Configure chassis network services for enhanced mode.

```
[edit chassis]
user@host>set network-services enhanced-ip
```

> **NOTE**: A router reboot might be required. See Network Services Mode Overview in the Junos OS Administration Library for details.

After you have completed the configuration, you can determine whether or not a Layer 3 VPN route is a part of a CNH by examining the display output of the following commands:

- `show route` *`route-value`* `extensive`
- `show route forwarding-table destination` *`destination-value`* `extensive`

Enabling Chained Composite Next Hops for IPv6-Labeled Unicast Routes

You can enable chained CNHs for IPv6-labeled unicast routes by configuring the *labeled-bgp* and *inet6* statements:

- To enable chained composite next hops for inet6 labeled unicast routes, include the *inet6* statement at the `[edit routing-options forwarding-table chained-composite-next-hop ingress labeled-bgp]` hierarchy level. This statement is disabled by default.

Example: Configuring Chained Composite Next Hops for Direct PE-PE Connections in VPNs

IN THIS SECTION

This example shows how to enable back-to-back Provider Edge (PE) router Layer 3 Virtual Private Network (VPN) connections with chained CNHs for MIC and MPC interfaces on MX Series and T4000 routers.

Requirements

This example uses the following hardware and software components:

- Six routers that can be a combination of MX240, MX480, MX960, or T4000 routers.
- Junos OS Release 13.3 running on all the devices.

Before you begin:

1. Configure the device interfaces.
2. Configure the following routing protocols on all the routers:
    a. MPLS
    b. BGP
    c. LDP LSPs as tunnels between the PE devices
    d. OSPF or any other IGP protocol

Overview

IN THIS SECTION

Prior to Junos OS Release 13.3, in a degenerated Layer 3 VPN case without the presence of an MPLS core router, previous behavior of flattened out indirect next hop and unicast next hop was utilized because there was no outer label available in the back-to-back PE-PE connection, and the ingress PE device only pushed single VPN labels. In a Layer 3 VPN multipath scenario with mixed PE-PE and PE-P-PE paths, chained CNHs could not be used either.

On platforms that support only MIC and MPC interfaces, chained CNHs are enabled by default. On platforms that support both DPC and MPC interfaces, the Layer 3 VPN configuration required the `pe-pe-connection` statement to support chained CNHs for PE-PE connections. However, the `pe-pe-connection` statement was not supported on platforms with MIC and FPC interfaces only.

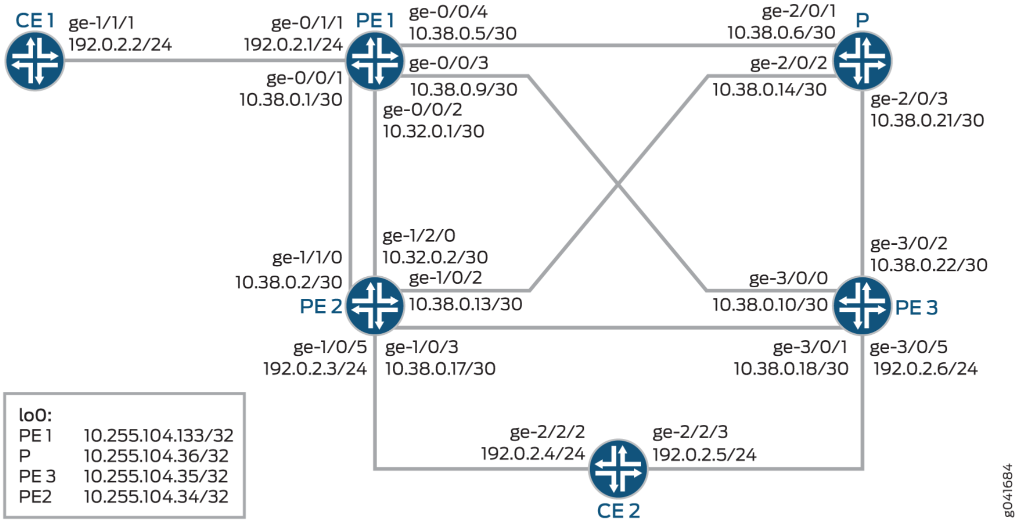
As a solution to these limitations, starting with Junos OS Release 13.3, the support for chained CNHs is enhanced to automatically identify the underlying platform capability on chainedCNHs at startup time, without relying on user configuration, and to decide the next-hop type (composite or indirect) to embed in the Layer 3 VPN label. This enhances the support for back-to-back PE-PE connections in Layer 3 VPN with chained CNHs, and eliminates the need for the `pe-pe-connection` statement.

To enable chained CNHs for directly connected PE devices, in addition to including the `l3vpn` statement at the `[edit routing-options forwarding-table chained-composite-next-hop ingress]` hierarchy level, make the following changes:

- On MX Series 5G Universal Routing Platforms containing both DPC and MPC FPCs, chained CNHs are disabled by default. To enable chained CNHs on the MX240, MX480, and MX960, the chassis must be configured to use the `enhanced-ip` option in network services mode.
- On T4000 Core Routers containing MPC and FPCs, chained CNHs are disabled by default. To enable chained CNHs on a T4000 router, the chassis must be configured to use the `enhanced-mode` option in network services mode.

Topology

Figure 92: Chained Composite Next Hops for PE-PE Connections



Configuration

IN THIS SECTION

CLI Quick Configuration

To quickly configure this example, copy the following commands, paste them into a text file, remove any line breaks, change any details necessary to match your network configuration, and then copy and paste the commands into the CLI at the `[edit]` hierarchy level.

CE1

```
set interfaces ge-1/1/1 unit 0 family inet address 192.0.2.2/24
set interfaces ge-1/1/1 unit 0 family iso
set interfaces ge-1/1/1 unit 0 family mpls
set interfaces lo0 unit 0 family inet address 198.51.100.1/24
set protocols bgp group PE type external
set protocols bgp group PE peer-as 200
set protocols bgp group PE neighbor 192.0.2.1
set routing-options autonomous-system 100
```

PE1

```
set interfaces ge-0/0/1 unit 0 family inet address 10.38.0.1/30
set interfaces ge-0/0/1 unit 0 family mpls
set interfaces ge-0/0/2 unit 0 family inet address 10.38.0.5/30
set interfaces ge-0/0/2 unit 0 family mpls
set interfaces ge-0/0/3 unit 0 family inet address 10.38.0.9/30
set interfaces ge-0/0/3 unit 0 family mpls
set interfaces ge-0/0/4 unit 0 family inet address 10.32.0.1/30
set interfaces ge-0/0/4 unit 0 family mpls
set interfaces ge-0/1/1 unit 0 family inet address 192.0.2.1/24
set interfaces ge-0/1/1 unit 0 family mpls
set interfaces lo0 unit 0 family inet address 10.255.104.133/32
set chassis network-services enhanced-ip
set routing-options forwarding-table chained-composite-next-hop ingress l3vpn
set routing-options autonomous-system 200
set routing-options forwarding-table export lbpp
set protocols mpls interface 10.38.0.1/30
set protocols mpls interface 10.32.0.1/30
set protocols mpls interface 10.38.0.5/30
set protocols mpls interface 10.38.0.9/30
set protocols bgp group PEs type internal
set protocols bgp group PEs local-address 10.255.104.133
set protocols bgp group PEs family inet unicast
set protocols bgp group PEs family inet-vpn unicast
set protocols bgp group PEs neighbor 10.255.104.134 local-preference 200
set protocols bgp group PEs neighbor 10.255.104.135
set protocols ospf area 0.0.0.0 interface all
set protocols ospf area 0.0.0.0 interface fxp0.0 disable
set protocols ospf area 0.0.0.0 interface lo0.0 passive
```

```
set protocols ldp interface all
set protocols ldp interface fxp0.0 disable
set policy-options policy-statement lbpp then load-balance per-packet
set routing-instances vpn-a instance-type vrf
set routing-instances vpn-a interface ge-0/1/1.0
set routing-instances vpn-a route-distinguisher 200:1
set routing-instances vpn-a vrf-target target:200:1
set routing-instances vpn-a vrf-table-label
set routing-instances vpn-a protocols bgp group CE type external
set routing-instances vpn-a protocols bgp group CE peer-as 100
set routing-instances vpn-a protocols bgp group CE neighbor 192.0.2.2
```

PE2

```
set interfaces ge-1/0/2 unit 0 family inet address 10.38.0.13/30
set interfaces ge-1/0/2 unit 0 family mpls
set interfaces ge-1/0/3 unit 0 family inet address 10.32.0.17/30
set interfaces ge-1/0/3 unit 0 family mpls
set interfaces ge-1/0/5 unit 0 family inet address 192.0.2.3/24
set interfaces ge-1/0/5 unit 0 family mpls
set interfaces ge-1/1/0 unit 0 family inet address 10.38.0.2/30
set interfaces ge-1/1/0 unit 0 family mpls
set interfaces ge-1/2/0 unit 0 family inet address 10.32.0.2/30
set interfaces ge-1/2/0 unit 0 family mpls
set interfaces lo0 unit 0 family inet address 10.255.104.134/32
set chassis network-services enhanced-ip
set routing-options forwarding-table chained-composite-next-hop ingress l3vpn
set routing-instances vpn-a instance-type vrf
set routing-instances vpn-a interface ge-1/0/5.0
set routing-instances vpn-a route-distinguisher 200:2
set routing-instances vpn-a vrf-target target:200:1
set routing-instances vpn-a protocols bgp group CE type external
set routing-instances vpn-a protocols bgp group CE peer-as 300
set routing-instances vpn-a protocols bgp group CE neighbor 192.0.2.3
set protocols mpls interface 10.38.0.2/30
set protocols mpls interface 10.32.0.2/30
set protocols mpls interface 10.38.0.13/30
set protocols mpls interface 10.38.0.17/30
set protocols bgp group PEs type internal
set protocols bgp group PEs local-address 10.255.104.134
set protocols bgp group PEs family inet unicast
set protocols bgp group PEs family inet-vpn unicast
```

```
set protocols bgp group PEs neighbor 10.255.104.133
set protocols bgp group PEs neighbor 10.255.104.135
set protocols ospf area 0.0.0.0 interface all
set protocols ospf area 0.0.0.0 interface fxp0.0 disable
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set protocols ldp interface all
set protocols ldp interface fxp0.0 disable
set routing-options autonomous-system 200
```

P

```
set interfaces ge-2/0/1 unit 0 family inet address 10.38.0.6/30
set interfaces ge-2/0/1 unit 0 family mpls
set interfaces ge-2/0/2 unit 0 family inet address 10.38.0.14/30
set interfaces ge-2/0/2 unit 0 family mpls
set interfaces ge-2/0/3 unit 0 family inet address 10.38.0.21/30
set interfaces ge-2/0/3 unit 0 family mpls
set interfaces lo0 unit 0 family inet address 10.255.104.136/32
set protocols mpls interface 10.38.0.6/30
set protocols mpls interface 10.38.0.14/30
set protocols mpls interface 10.38.0.21/30
set protocols bgp group PEs type internal
set protocols bgp group PEs local-address 10.255.104.136
set protocols bgp group PEs family inet unicast
set protocols bgp group PEs family inet-vpn unicast
set protocols bgp group PEs neighbor 10.255.104.133
set protocols ospf area 0.0.0.0 interface all
set protocols ospf area 0.0.0.0 interface fxp0.0 disable
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set protocols ldp interface all
set protocols ldp interface fxp0.0 disable
set routing-options autonomous-system 200
```

PE3

```
set interfaces ge-3/0/0 unit 0 family inet address 10.38.0.10/30r0-r3
set interfaces ge-3/0/0 unit 0 family mpls
set interfaces ge-3/0/1 unit 0 family inet address 10.38.0.18/30r0-r1-2
set interfaces ge-3/0/1 unit 0 family mpls
set interfaces ge-3/0/2 unit 0 family inet address 10.38.0.22/30
set interfaces ge-3/0/2 unit 0 family mpls
```

```
set interfaces ge-3/0/5 unit 0 family inet address 192.0.2.6/24r0-r1-1
set interfaces ge-3/0/5 unit 0 family mpls
set interfaces lo0 unit 0 family inet address 10.255.104.135/32
set chassis network-services enhanced-mode
set routing-options forwarding-table chained-composite-next-hop ingress l3vpn
set routing-options autonomous-system 200
set routing-instances vpn-a instance-type vrf
set routing-instances vpn-a interface ge-3/0/5.0
set routing-instances vpn-a route-distinguisher 200:3
set routing-instances vpn-a vrf-target target:200:1
set routing-instances vpn-a protocols bgp group CE type external
set routing-instances vpn-a protocols bgp group CE peer-as 300
set routing-instances vpn-a protocols bgp group CE neighbor 192.0.2.5
set protocols mpls interface 10.38.0.10/30
set protocols mpls interface 10.38.0.18/30
set protocols mpls interface 10.38.0.22/30
set protocols bgp group PEs type internal
set protocols bgp group PEs local-address 10.255.104.135
set protocols bgp group PEs family inet unicast
set protocols bgp group PEs family inet-vpn unicast
set protocols bgp group PEs neighbor 10.255.104.133
set protocols bgp group PEs neighbor 10.255.104.134
set protocols ospf area 0.0.0.0 interface all
set protocols ospf area 0.0.0.0 interface fxp0.0 disable
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set protocols ldp interface all
set protocols ldp interface fxp0.0 disable
```

CE2

```
set interfaces ge-2/2/2 unit 0 family inet address 192.0.2.4/24
set interfaces ge-2/2/2 unit 0 family mpls
set interfaces ge-2/2/3 unit 0 family inet address 192.0.2.5/24
set interfaces ge-2/2/3 unit 0 family mpls
set interfaces lo0 unit 0 family inet address 198.51.100.2/24
set protocols bgp group PE type external
set protocols bgp group PE metric-out 50
set protocols bgp group PE peer-as 200
set protocols bgp group PE export s2b
set protocols bgp group PE neighbor 192.0.2.4
set protocols bgp group PE neighbor 192.0.2.5
set policy-options policy-statement s2b from protocol direct
```

```
set policy-options policy-statement s2b then accept
set routing-options autonomous-system 300
```

Configuring Multipath Layer 3 VPN with Chained Composite Next Hops

Step-by-Step Procedure

The following example requires you to navigate various levels in the configuration hierarchy. For information about navigating the CLI, see *Using the CLI Editor in Configuration Mode*.

To configure basic Layer 3 VPN with chained CNH on the PE1 router:

NOTE: Repeat this procedure for the PE2 and PE3 routers in the MPLS domain, after modifying the appropriate interface names, addresses, and any other parameters for each router.

1. Configure the interfaces on the PE1 router.

```
PE1 to CE1
[edit interfaces]
user@PE1 # set ge-0/1/1 unit 0 family inet address 192.0.2.1/24
user@PE1 # set ge-0/1/1 unit 0 family mpls
```

```
PE1 to PE2
[edit interfaces]
user@PE1 # set ge-0/0/1 unit 0 family inet address 10.38.0.1/30
user@PE1 # set ge-0/0/1 unit 0 family mpls
user@PE1 # set ge-0/0/2 unit 0 family inet address 10.38.0.5/30
user@PE1 # set ge-0/0/2 unit 0 family mpls
```

```
PE1 to P
[edit interfaces]
user@PE1 # set ge-0/0/4 unit 0 family inet address 10.32.0.1/30
user@PE1 # set ge-0/0/4 unit 0 family mpls
```

```
PE1 to PE3
[edit interfaces]
```

```
user@PE1 # set ge-0/0/3 unit 0 family inet address 10.38.0.9/30
user@PE1 # set ge-0/0/3 unit 0 family mpls
```

```
Loopback interface
[edit interfaces]
user@PE1 # set lo0 unit 0 family inet address 10.255.104.133/32
```

2. Enable enhanced IP mode on the PE1 chassis.

```
[edit chassis]
use@PE1# set network-services enhanced-ip
```

3. Enable chained CNH on the global Layer 3 VPN.

```
[edit routing-options]
use@PE1# set forwarding-table chained-composite-next-hop ingress l3vpn
```

4. Configure the autonomous system for PE1.

```
[edit routing-options]
user@PE1# set autonomous-system 200
```

5. Export the policy configured for load balancing.

```
[edit routing-options]
user@PE1# set forwarding-table export lbpp
```

6. Configure MPLS on the PE1 interfaces connecting to the P router and other PE routers.

```
[edit protocols]
user@PE1# set mpls interface 10.38.0.1/30
user@PE1# set mpls interface 10.32.0.1/30
user@PE1# set mpls interface 10.38.0.5/30
user@PE1# set mpls interface 10.38.0.9/30
```

7. Configure the IBGP group for PE1 to peer with the PE2 and PE3 routers.

```
[edit protocols]
user@PE1# set bgp group PEs type internal
user@PE1# set bgp group PEs local-address 10.255.104.133
user@PE1# set bgp group PEs family inet unicast
user@PE1# set bgp group PEs family inet-vpn unicast
user@PE1# set bgp group PEs neighbor 10.255.104.134 local-preference 200
user@PE1# set bgp group PEs neighbor 10.255.104.135
```

8. Configure OSPF with traffic engineering capability on all the interfaces of PE1, excluding the management interface.

```
[edit protocols]
user@PE1# set ospf area 0.0.0.0 interface all
user@PE1# set ospf area 0.0.0.0 interface fxp0.0 disable
user@PE1# set ospf area 0.0.0.0 interface lo0.0 passive
```

9. Configure LDP on all the interfaces of PE1, excluding the management interface.

```
[edit protocols]
user@PE1# set ldp interface all
user@PE1# set ldp interface fxp0.0 disable
```

10. Configure a policy to load-balance traffic on a per packet basis.

```
[edit policy-options]
user@PE1# set policy-statement lbpp then load-balance per-packet
```

11. Configure a VRF routing instance on the CE1-facing interface of PE1.

```
[edit routing-instances]
user@PE1# set vpn-a instance-type vrf
user@PE1# set vpn-a interface ge-0/1/1.0
```

12. Configure the routing instance parameters.

```
[edit routing-instances]
user@PE1# set vpn-a route-distinguisher 200:1
user@PE1# set vpn-a vrf-target target:200:1
user@PE1# set vpn-a vrf-table-label
```

13. Configure an EBGP group for the routing instance, so PE1 can peer with CE1.

```
[edit routing-instances]
user@PE1# set vpn-a protocols bgp group CE type external
user@PE1# set vpn-a protocols bgp group CE peer-as 100
user@PE1# set vpn-a protocols bgp group CE neighbor 192.0.2.2
```

Results

From configuration mode, confirm your configuration by entering the `show chassis`, `show interfaces`, `show protocols`, `show routing-options`, `show routing-instances`, and `show policy-options` commands. If the output does not display the intended configuration, repeat the instructions in this example to correct the configuration.

PE1

```
user@PE1# show chassis
network-services enhanced-ip;
```

```
user@PE1# show interfaces
ge-0/0/1 {
    unit 0 {
        family inet {
            address 10.38.0.1/30;
        }
        family mpls;
    }
}
ge-0/0/2 {
    unit 0 {
        family inet {
            address 10.38.0.5/30;
```

```
        }
        family mpls;
    }
}
ge-0/0/3 {
    unit 0 {
        family inet {
            address 10.38.0.9/30;
        }
        family mpls;
    }
}
ge-0/0/4 {
    unit 0 {
        family inet {
            address 10.32.0.1/30;
        }
        family mpls;
    }
}
ge-0/1/1 {
    unit 0 {
        family inet {
            address 192.0.2.1/24;
        }
        family mpls;
    }
}
lo0 {
    unit 0 {
        family inet {
            address 10.255.104.133/32;
        }
    }
}
```

```
user@PE1# show protocols
mpls {
    interface 10.38.0.1/30;
    interface 10.32.0.1/30;
    interface 10.38.0.5/30;
```

```
    interface 10.38.0.9/30;
}
bgp {
    group PEs {
        type internal;
        local-address 10.255.104.133;
        family inet {
            unicast;
        }
        family inet-vpn {
            unicast;
        }
        neighbor 10.255.104.134 {
            local-preference 200;
        }
        neighbor 10.255.104.135;
    }
}
ospf {
    area 0.0.0.0 {
        interface all;
        interface fxp0.0 {
            disable;
        }
        interface lo0.0 {
            passive;
        }
    }
}
ldp {
    interface all;
    interface fxp0.0 {
        disable;
    }
}
```

```
user@PE1# show routing-options
autonomous-system 200;
forwarding-table {
    export lbpp;
    chained-composite-next-hop {
```

```
        ingress {
            l3vpn;
        }
    }
}
```

```
user@PE1# show routing-instances
vpn-a {
    instance-type vrf;
    interface ge-0/1/1.0;
    route-distinguisher 200:1;
    vrf-target target:200:1;
    vrf-table-label;
    protocols {
        bgp {
            group CE {
                type external;
                peer-as 100;
                neighbor 192.0.2.2;
            }
        }
    }
}
```

```
user@PE1# show policy-options
policy-statement lbpp {
    then {
        load-balance per-packet;
    }
}
```

Verification

IN THIS SECTION

Confirm that the configuration is working properly.

Verifying the Routes

Purpose

Verify that the Layer 3 VPN prefixes toward PE1-PE2 point to chained CNHs.

Action

From operational mode, run the `show route 198.51.100.2 table vpn-a extensive` command.

```
user@PE1> show route 198.51.100.2 table vpn-a extensive


  vpn-a.inet.0: 7 destinations, 10 routes (7 active, 0 holddown, 0 hidden)
  198.51.100.2/24 (2 entries, 1 announced)
  TSI:
  KRT in-kernel 198.51.100.2/3 -> {composite(720)}
  Page 0 idx 0, (group CE type External) Type 1 val 938eaa8 (adv_entry)
     Advertised metrics:
       Nexthop: Self
       AS path: [200] 300 I
       Communities: target:200:1
  Path 198.51.100.2 from 10.255.104.133 Vector len 4.  Val: 0
          *BGP    Preference: 170/-101
                  Route Distinguisher: 200:2
                  Next hop type: Indirect
                  Address: 0x9391654
                  Next-hop reference count: 12
                  Source: 10.255.104.133
                  Next hop type: Router, Next hop index: 1048580
                  Next hop: 10.32.0.2 via ge-0/0/2.0
                  Session Id: 0x1
                  Next hop: 10.38.0.2 via ge-0/0/1.0, selected
                  Session Id: 0x3
                  Protocol next hop: 10.255.104.133
                  Push 300192
                  Composite next hop: 0x93918a4 718 INH Session ID: 0x9
                  Indirect next hop: 0x941c000 1048581 INH Session ID: 0x9
                  State: <Secondary Active Int Ext ProtectionCand>
                  Local AS:   200 Peer AS:   200
```

```
                Age: 28   Metric: 50      Metric2: 1
                Validation State: unverified
                Task: BGP_203.0.113.1.133+57173
                Announcement bits (2): 0-KRT 1-BGP_RT_Background
                AS path: 300 I
                Communities: target:200:1
                Import Accepted
                VPN Label: 300192
                Localpref: 100
                Router ID: 10.255.104.133
                Primary Routing Table bgp.l3vpn.0
                Composite next hops: 1
                        Protocol next hop: 10.255.104.133 Metric: 1
                        Push 300192
                        Composite next hop: 0x93918a4 718 INH Session ID: 0x9
                        Indirect next hop: 0x941c000 1048581 INH Session ID: 0x9
                        Indirect path forwarding next hops: 2
                                Next hop type: Router
                                Next hop: 10.32.0.2 via ge-1/0/0.0
                                Session Id: 0x1
                                Next hop: 10.38.0.2 via ge-1/1/2.0
                                Session Id: 0x3
                  10.255.104.133/32 Originating RIB: inet.3
                    Metric: 1                       Node path count: 1
                    Forwarding nexthops: 2
                          Nexthop: 10.32.0.2 via ge-0/0/2.0
         BGP    Preference: 170/-101
                Route Distinguisher: 200:3
                Next hop type: Indirect
                Address: 0x9391608
                Next-hop reference count: 9
                Source: 10.255.104.131
                Next hop type: Router, Next hop index: 722
                Next hop: 10.38.0.10 via ge-0/0/1.0, selected
                Session Id: 0x4
                Protocol next hop: 10.255.104.131
                Push 299936
                Composite next hop: 0x9391690 723 INH Session ID: 0xb
                Indirect next hop: 0x941c0fc 1048583 INH Session ID: 0xb
                State: <Secondary NotBest Int Ext ProtectionCand>
                Inactive reason: Not Best in its group - Router ID
                Local AS:   200 Peer AS:   200
                Age: 28   Metric: 50      Metric2: 1
```

```
                    Validation State: unverified
                    Task: BGP_203.0.113.1.131+63797
                    AS path: 300 I
                    Communities: target:200:1
                    Import Accepted
                    VPN Label: 299936
                    Localpref: 100
                    Router ID: 10.255.104.131
                    Primary Routing Table bgp.l3vpn.0
                    Composite next hops: 1
                            Protocol next hop: 10.255.104.131 Metric: 1
                            Push 299936
                            Composite next hop: 0x9391690 723 INH Session ID: 0xb
                            Indirect next hop: 0x941c0fc 1048583 INH Session ID: 0xb
                            Indirect path forwarding next hops: 1
                                    Next hop type: Router
                                    Next hop: 10.38.0.10 via ge-1/0/2.0
                                    Session Id: 0x4
                      10.255.104.131/32 Originating RIB: inet.3
                        Metric: 1                       Node path count: 1
                        Forwarding nexthops: 1
                              Nexthop: 10.38.0.10 via ge-1/0/2.0
```

Meaning

The PE2 router is the CNH for PE1 to reach CE2.

Verifying Chained Next Hops on Direct PE-PE Connection

Purpose

Verify that chained next hop is generated for direct PE-PE connection on CE1.

Action

From operational mode, run the `ping` command.

```
user@CE1> ping 192.0.2.4
!!!!!
```

```
--- lsping statistics ---
5 packets transmitted, 5 packets received, 0% packet loss
```

Meaning

Chained CNH is enabled for the PE1 to PE2 connection.

RELATED DOCUMENTATION

- Chained Composite Next Hops for Transit Devices for VPNs
- Allocating More Memory for Routing Tables, Firewall Filters, and Layer 3 VPN Labels
- Network Services Mode Overview
- Configuring Junos OS to Run a Specific Network Services Mode in MX Series Routers
- *chained-composite-next-hop*
- transit (Chained Composite Next Hops)
- *ingress*

Class of Service for VPNs

IN THIS SECTION

VPNs and Class of Service

You can configure Junos class-of-service (CoS) features to provide multiple classes of service for VPNs. The CoS features are supported on Layer2 VPNs, Layer 3 VPNs, and VPLS. On the router, you can configure multiple forwarding classes for transmitting packets, define which packets are placed into each output queue, schedule the transmission service level for each queue, and manage congestion using a random early detection (RED) algorithm.

VPNs use the standard CoS configuration.

Rewriting Class of Service Markers and VPNs

A marker reads the current forwarding class and loss priority information associated with a packet and finds the chosen code point from a table. It then writes the code point information into the packet header. Entries in a marker configuration represent the mapping of the current forwarding class into a new forwarding class, to be written into the header.

You define markers in the *rewrite rules* section of the class-of-service (CoS) configuration hierarchy and reference them in the *logical interface* configuration. You can configure different rewrite rules to handle VPN traffic and non-VPN traffic. The rewrite rule can be applied to MPLS and IPv4 packet headers simultaneously, making it possible to initialize MPLS experimental (EXP) and IP precedence bits at LSP ingress.

For a detailed example of how to configure rewrite rules for MPLS and IPv4 packets and for more information about how to configure statements at the `[edit class-of-service]` hierarchy level, see the Class of Service User Guide (Routers and EX9200 Switches).

Configuring Traffic Policing in Layer 3 VPNs

You can use policing to control the amount of traffic flowing over the interfaces servicing a Layer 3 VPN. If policing is disabled on an interface, all the available bandwidth on a Layer 3 VPN tunnel can be used by a single *CCC* or *TCC* interface.

For more information about the `policer` statement, see the Routing Policies, Firewall Filters, and Traffic Policers User Guide.

To enable Layer 3 VPN policing on an interface, include the `policer` statement:

```
policer {
    input policer-template-name;
    output policer-template-name;
}
```

If you configure CCC encapsulation, you can include the `policer` statement at the following hierarchy levels:

- `[edit interfaces interface-name unit logical-unit-number family ccc]`
- `[edit logical-systems logical-system-name interfaces interface-name unit logical-unit-number family ccc]`

If you configure TCC encapsulation, you can include the `policer` statement at the following hierarchy levels:

- `[edit interfaces interface-name unit logical-unit-number family tcc]`

- `[edit logical-systems logical-system-name interfaces interface-name unit logical-unit-number family tcc]`

SEE ALSO

> Routing Policies, Firewall Filters, and Traffic Policers User Guide

Applying Custom MPLS EXP Classifiers to Routing Instances in Layer 3 VPNs

When you include the `vrf-table-label` statement in the configuration for a routing instance (as described in "Filtering Packets in Layer 3 VPNs Based on IP Headers" on page 55) but do not explicitly apply a classifier to the routing instance, the default MPLS EXP classifier is applied.

For PICs that are installed on Enhanced FPCs, you can apply a custom classifier to override the default MPLS EXP classifier for the routing instance. For detailed instructions, see the Class of Service User Guide (Routers and EX9200 Switches). The following instructions serve as a summary:

1. Filter traffic based on the IP header by including the `vrf-table-label` statement at the `[edit routing-instances routing-instance-name]` hierarchy level:

```
[edit routing-instances routing-instance-name]
vrf-table-label;
```

2. Configure a custom MPLS EXP classifier by including the appropriate statements at the `[edit class-of-service]` hierarchy level. For instructions, see the Class of Service User Guide (Routers and EX9200 Switches).

3. Configure the routing instance for CoS by including the `routing-instances` statement at the `[edit class-of-service]` hierarchy level:

```
[edit class-of-service]
routing-instances routing-instance-name {
    classifiers {
        exp (classifier-name | default);
    }
}
```

4. Configure the routing instance to use the custom MPLS EXP classifier by including the `classifiers` statement at the `[edit class-of-service routing-instances` *`routing-instance-name`*`]` hierarchy level:

```
[edit class-of-service routing-instances routing-instance-name]
classifiers {
    exp classifier-name;
}
```

To display the MPLS EXP classifiers associated with all routing instances, issue the `show class-of-service routing-instances` command.

NOTE: The following caveats apply to custom MPLS EXP classifiers for routing instances:

- An Enhanced FPC is required.
- Logical systems are not supported.

RELATED DOCUMENTATION

Class of Service User Guide (Routers and EX9200 Switches)

Graceful Restarts for VPNs

IN THIS SECTION

VPN Graceful Restart

IN THIS SECTION

With routing protocols, any service interruption requires that an affected router recalculate adjacencies with neighboring routers, restore routing table entries, and update other protocol-specific information. An unprotected restart of the router results in forwarding delays, route flapping, wait times stemming from protocol reconvergence, and even dropped packets. Graceful restart allows a routing device undergoing a restart to inform its adjacent neighbors and peers of its condition. During a graceful restart, the restarting device and its neighbors continue forwarding packets without disrupting network performance.

For VPN graceful restart to function properly, the following items need to be configured on the PE router:

- BGP graceful restart must be active on the PE-to-PE sessions carrying any service-signaling data in the session’s network layer reachability information (NLRI).
- OSPF, IS-IS, LDP, and RSVP graceful restart must be active, because routes added by these protocols are used to resolve VPN NLRIs.
- For other protocols (static, Routing Information Protocol [RIP], and so on), graceful restart functionality must also be active when these protocols are run between the PE and CE routers. Layer 2 VPNs do not rely on this, because protocols are not configured between the PE and CE routers.

In VPN graceful restart, a restarting router completes the following procedures:

- Waits for all the BGP NLRI information from other PE routers before it starts advertising routes to its CE routers.
- Waits for all protocols in all routing instances to converge (or finish graceful restart) before sending CE router information to the other PE routers.
- Waits for all routing instance information (whether it is local configuration or advertisements from a remote peer router) to be processed before sending it to the other PE routers.
- Preserves all forwarding state information in the MPLS routing tables until new labels and transit routes are allocated and then advertises them to other PE routers (and CE routers in carrier-of-carriers VPNs).

Graceful restart is supported on Layer 2 VPNs, Layer 3 VPNs, and virtual-router routing instances.

Benefit of a VPN graceful restart

The main benefit of a VPN graceful restart is that it allows a router whose VPN control plane is undergoing a restart to continue to forward traffic while recovering its state from neighboring routers. It temporarily suppresses all routing protocol updates and enables a router to pass through intermediate convergence states that are hidden from the rest of the network. Without graceful restart, a control plane restart disrupts the VPN services provided by the router.

Configuring Graceful Restart for VPNs

You can configure graceful restart to enable a router to pass through intermediate convergence states that are hidden from the rest of the network. Graceful restart allows a router whose VPN control plane is undergoing a restart (restarting router) to continue to forward traffic while recovering its state from neighboring routers (helper routers).

The restarting router requests a grace period from the neighbor or peer, which can then cooperate with the restarting router. When a restart event occurs and graceful restart is enabled, the restarting router can still forward traffic during the restart period, and convergence in the network is not disrupted. The helper routers hide the restart event from other devices not directly connected to the restarting router. In other words, the restart is not visible to the rest of the network, and the restarting router is not removed from the network topology.

Without graceful restart, a control plane restart disrupts any VPN services provided by the router. Graceful restart is supported on Layer 2 VPNs, Layer 3 VPNs, virtual-router routing instances, and VPLS.

The graceful restart request occurs only if the following conditions are met:

- The network topology is stable.
- The neighbor or peer routers cooperate.
- The restarting router is not already cooperating with another restart already in progress.
- The grace period does not expire.

Before you begin:

- Configure the devices for network communication.
- Configure the device interfaces.

Graceful restart is disabled by default. To enable VPN graceful restart:

1. Configure graceful restart globally.

   ```
   [edit routing-options]
   user@host# set graceful-restart
   ```

   NOTE:

   - Graceful restart can be enabled on logical systems. To configure graceful restart globally, include the `graceful-restart` statement at the `[edit logical-systems logical-system-name routing-options]` or the `[edit logical-systems logical-system-name routing-instances routing-instance-name routing-options]` hierarchy levels.

   - To disable graceful restart globally, include the `disable` statement at the `[edit routing-options graceful-restart]` hierarchy level.

     For example:

     ```
     [edit routing-options]
     user@host# set graceful-restart disable
     ```

2. Enable or disable graceful restart on a per-protocol, per-group, or per-neighbor basis, depending on the specific protocol, where the most specific definition is used.

   ```
   [edit protocols]
   user@host# set bgp graceful-restart
   user@host# set bgp group group-name type internal local-address local-ip-address neighbor
   neighbor1-address
   user@host# set bgp group group-name type internal local-address local-ip-address neighbor
   neighbor2-address graceful-restart disable
   ```

3. Configure graceful restart for Layer 3 VPNS for all routing and MPLS-related protocols within a routing instance. Because you can configure multi-instance BGP and multi-instance LDP, graceful restart for a carrier-of-carriers scenario is supported.

   ```
   [edit routing-instance]
   user@host# set routing-instance-name routing-options graceful-restart
   ```

NOTE:

- To disable graceful restart globally, include the `disable` statement at the `[edit routing-instances *routing-instance-name* routing-options graceful-restart]` hierarchy level.

  For example:

  ```
  [edit routing-instances]
  user@host# set instance1 routing-options graceful-restart disable
  ```

- To disable graceful restart for individual protocols, include the `disable` statement at the `[edit routing-instances *routing-instance-name* protocols *protocol-name* graceful-restart]` hierarchy level.

  For example:

  ```
  [edit routing-instances]
  user@host# set instance1 protocols ospf graceful-restart disable
  ```

4. Configure the duration of the graceful restart period for the routing instance.

   ```
   [edit routing-options]
   user@host# set graceful-restart restart-duration seconds
   ```

   The `restart-duration` option sets the period of time that the router waits for a graceful restart to be completed. You can configure a time between 1 through 600 seconds. The default value is 300 seconds. At the end of the configured time period, the router performs a standard restart without recovering its state from the neighboring routers. This disrupts VPN services, but is probably necessary if the router is not functioning normally.

   NOTE: You can include the `restart-duration` option at either the global or routing instance level.

Configuring Nonstop Active Routing for BGP Multicast VPN

BGP multicast virtual private network (MVPN) is a Layer 3 VPN application that is built on top of various unicast and multicast routing protocols such as Protocol Independent Multicast (PIM), BGP, RSVP, and LDP. Enabling nonstop active routing (NSR) for BGP MVPN requires that NSR support is enabled for all these protocols.

Before you begin:

- Configure the router interfaces. See Interfaces Fundamentals.
- Configure an interior gateway protocol or static routing. See the Junos OS Routing Protocols Library.
- Configure a multicast group membership protocol (IGMP or MLD). See Understanding IGMP and Understanding MLD.
- For this feature to work with IPv6, the routing device must be running Junos OS Release 10.4 or later.

The state maintained by MVPN includes MVPN routes, cmcast, provider-tunnel, and forwarding information. BGP MVPN NSR synchronizes this MVPN state between the primary and backup Routing Engines. While some of the state on the backup Routing Engine is locally built based on the configuration, most of it is built based on triggers from other protocols that MVPN interacts with. The triggers from these protocols are in turn the result of state replication performed by these modules. This includes route change notifications by unicast protocols, join and prune triggers from PIM, remote MVPN route notification by BGP, and provider-tunnel related notifications from RSVP and LDP.

Configuring NSR and unified in-service software upgrade (ISSU) support to the BGP MVPN protocol provides features such as various provider tunnel types, different MVPN modes (source tree, shared-tree), and PIM features. As a result, at the ingress PE, replication is turned on for dynamic LSPs. Thus, when NSR is configured, the state for dynamic LSPs is also replicated to the backup Routing Engine. After the state is resolved on the backup Routing Engine, RSVP sends required notifications to MVPN.

To enable BGP MVPN NSR support, the `advertise-from-main-vpn-tables` configuration statement needs to be configured at the `[edit protocols bgp]` hierarchy level.

Nonstop active routing configurations include two Routing Engines that share information so that routing is not interrupted during Routing Engine failover. When NSR is configured on a dual Routing Engine platform, the PIM control state is replicated on both Routing Engines.

This PIM state information includes:

- Neighbor relationships
- Join and prune information
- RP-set information
- Synchronization between routes and next hops and the forwarding state between the two Routing Engines

Junos OS supports NSR in the following PIM scenarios:

- Dense mode

- Sparse mode
- SSM
- Static RP
- Auto-RP (for IPv4 only)
- Bootstrap router
- Embedded RP on the non-RP router (for IPv6 only)
- BFD support
- Draft Rosen multicast VPNs and BGP multicast VPNs
- Policy features such as neighbor policy, bootstrap router export and import policies, scope policy, flow maps, and reverse path forwarding (RPF) check policies

To configure nonstop active routing:

1. Because NSR requires you to configure graceful Routing Engine switchover (GRES), to enable GRES, include the `graceful-switchover` statement at the `[edit chassis redundancy]` hierarchy level.

   ```
   [edit]
   user@host# set chassis redundancy graceful-switchover
   ```

2. Include the `synchronize` statement at the `[edit system]` hierarchy level so that configuration changes are synchronized on both Routing Engines.

   ```
   [edit system]
   user@host# set synchronize
   user@host# exit
   ```

3. Configure PIM settings on the desingated routerwith sparse mode and version, and static address pointing to the rendezvous points.

   ```
   [edit protocols pim]
   user@host# set rp static address address
   user@host# set interface interface-name mode sparse
   user@host# set interface interface-name version 2
   ```

For example, to set sparse mode, version 2 and static address:

```
[edit protocols pim]
user@host# set rp static address 10.210.255.202
user@host# set interface fe-0/1/3.0 mode sparse
user@host# set interface fe-0/1/3.0 version 2
```

4. Configure per-packet load balancing on the designated router.

```
[edit policy-options policy-statement policy-name]
user@host# set then policy-name per-packet
```

For example, to set load-balance policy:

```
[edit policy-options policy-statement load-balance]
user@host# set then load-balance per-packet
```

5. Apply the load-balance policy on the designated router.

```
[edit]
user@host# set routing-options forwarding-table export load-balance
```

6. Configure nonstop active routing on the designated router.

```
[edit]
user@host# set routing-options nonstop-routing
user@host# set routing-options router-id address
```

For example, to set nonstop active routing on the designated router with address 10.210.255.201:

```
[edit]
user@host# set routing-options router-id 10.210.255.201
```

SEE ALSO

Configuring Basic PIM Settings

Understanding Nonstop Active Routing for PIM

RELATED DOCUMENTATION

Graceful Restart and Layer 2 and Layer 3 VPNs
Graceful Restart System Requirements
Graceful Restart Concepts
Verifying Graceful Restart Operation

CHAPTER 8

Troubleshooting Layer 3 VPNs

IN THIS CHAPTER

Pinging VPNs

IN THIS SECTION

Pinging VPNs, VPLS, and Layer 2 Circuits

For testing purposes, you can ping Layer 2 VPNs, Layer 3 VPNs, and Layer 2 circuits by using the `ping mpls` command. The `ping mpls` command helps to verify that a VPN or circuit has been enabled and tests the integrity of the VPN or Layer 2 circuit connection between the PE routers. It does not test the connection between a PE router and a CE router. To ping a VPLS routing instance, you issue a `ping vpls instance` command (see "Pinging a VPLS Routing Instance" on page 1225).

You issue the `ping mpls` command from the ingress PE router of the VPN or Layer 2 circuit to the egress PE router of the same VPN or Layer 2 circuit. When you execute the `ping mpls` command, echo requests are sent as MPLS packets.

The payload is a User Datagram Protocol (UDP) packet forwarded to the address `127.0.0.1`. The contents of this packet are defined in RFC 4379, *Detecting Multi-Protocol Label Switched (MPLS) Data Plane*

Failures. The label and interface information for building and sending this information as an MPLS packet is the same as for standard VPN traffic, but the time-to-live (TTL) of the innermost label is set to 1.

When the echo request arrives at the egress PE router, the contents of the packet are checked, and then a reply that contains the correct return is sent by means of UDP. The PE router sending the echo request waits to receive an echo reply after a timeout of 2 seconds (you cannot configure this value).

You must configure MPLS at the `[edit protocols mpls]` hierarchy level on the egress PE router (the router receiving the MPLS echo packets) to be able to ping the VPN or Layer 2 circuit. You must also configure the address `127.0.0.1/32` on the egress PE router's `lo0` interface. If this is not configured, the egress PE router does not have this forwarding entry and therefore simply drops the incoming MPLS pings.

The `ping mpls` command has the following limitations:

- You cannot ping an IPv6 destination prefix.
- You cannot ping a VPN or Layer 2 circuit from a router that is attempting a graceful restart.
- You cannot ping a VPN or Layer 2 circuit from a logical system.

You can also determine whether an LSP linking two PE routers in a VPN is up by pinging the end point address of the LSP. The command you use to ping an MPLS LSP end point is `ping mpls lsp-end-point` *`address`*. This command tells you what type of LSP (RSVP or LDP) terminates at the address specified and whether that LSP is up or down.

For a detailed description of this command, see the *Junos Routing Protocols and Policies Command Reference*.

Setting the Forwarding Class of the Ping Packets

When you execute the `ping mpls` command, the ping packets forwarded to the destination include MPLS labels. It is possible to set the value of the forwarding class for these ping packets by using the **exp** option with the `ping mpls` command. For example, to set the forwarding class to 5 when pinging a Layer 3 VPN, issue the following command:

```
ping mpls l3vpn westcoast source 192.0.2.0 prefix 192.0.2.1 exp 5 count 20 detail
```

This command would makes the router attempt to ping the Layer 3 VPN `westcoast` using ping packets with an EXP forwarding class of 5. The default forwarding class used for the `ping mpls` command packets is 7.

Pinging a VPLS Routing Instance

The `ping vpls instance` command uses a different command structure and operates in a different fashion than the `ping mpls` command used for VPNs and Layer 2 circuits. The `ping vpls instance` command is only supported on MX Series routers, the M120 router, the M320 router, and the T1600 router.

To ping a VPLS routing instance, use the following command:

```
ping vpls instance instance-name destination-mac address source-ip address <count number> <data-
plane-response> <detail> <learning-vlan-id number> <logical-system logical-system-name>
```

Pinging a VPLS routing instance requires using the `ping vpls instance` command with a combination of the routing instance name, the destination MAC address, and the source IP address (IP address of the outgoing interface).

When you run this command, you are provided feedback on the status of your request. An exclamation point (!) indicates that an echo reply was received. A period (`.`) indicates that an echo reply was not received within the timeout period. An `x` indicates that an echo reply was received with an error code these packets are not counted in the received packets count. They are accounted for separately.

Use Feature Explorer to confirm platform and release support for `ping vpls instance`. For more details, including argument descriptions and additional options, see ping vpls instance.

Pinging a Layer 3 VPN

To ping a Layer 3 VPN, use the following command:

```
ping mpls l3vpn l3vpn-name prefix prefix <count count>
```

You ping a combination of an IPv4 destination prefix and a Layer 3 VPN name on the egress PE router to test the integrity of the VPN connection between the ingress and egress PE routers. The destination prefix corresponds to a prefix in the Layer 3 VPN. However, the ping tests only whether the prefix is present in a PE router’s VRF table. It does not test the connection between a PE router and a CE router.

Troubleshooting Layer 3 VPNs

IN THIS SECTION

Diagnosing Common Layer 3 VPN Problems

IN THIS SECTION

Problem

Description

To troubleshoot problems in the Layer 3 VPN configuration, start at one end of the VPN (the local customer edge [CE] router) and follow the routes to the other end of the VPN (the remote CE router).

Solution

The following troubleshooting steps should help you diagnose common problems:

1. If you configured a routing protocol between the local provider edge (PE) and CE routers, check that the peering and adjacency are fully operational. When you do this, be sure to specify the name of the routing instance. For example, to check OSPF adjacencies, enter the `show ospf neighbor instance` *`routing-instance-name`* command on the PE router.

   If the peering and adjacency are not fully operational, check the routing protocol configuration on the CE router and check the routing protocol configuration for the associated VPN routing instance on the PE router.

2. Check that the local CE and PE routers can ping each other.

   To check that the local CE router can ping the VPN interface on the local PE router, use a `ping` command in the following format, specifying the IP address or name of the PE router:

   ```
   user@host> ping (ip-address | host-name)
   ```

To check that the local PE router can ping the CE router, use a `ping` command in the following format, specifying the IP address or name of the CE router, the name of the interface used for the VPN, and the source IP address (the local address) in outgoing echo request packets:

```
user@host> ping ip-address interface interface local echo-address
```

Often, the peering or adjacency between the local CE and local PE routers must come up before a `ping` command is successful. To check that a link is operational in a lab setting, remove the interface from the VPN routing and forwarding (VRF) by deleting the `interface` statement from the `[edit routing-instance` *`routing-instance-name`*`]` hierarchy level and recommitting the configuration. Doing this removes the interface from the VPN. Then try the **ping** command again. If the command is successful, configure the interface back into the VPN and check the routing protocol configuration on the local CE and PE routers again.

3. On the local PE router, check that the routes from the local CE router are in the VRF table (*routing-instance-name*.inet.0):

```
user@host> show route table routing-instance-name.inet.0 <detail>
```

The following example shows the routing table entries. Here, the loopback address of the CE router is `10.255.14.155/32` and the routing protocol between the PE and CE routers is BGP. The entry looks like any ordinary BGP announcement.

```
10.255.14.155/32 (1 entry, 1 announced)
        *BGP    Preference: 170/-101
                Nexthop: 192.168.197.141 via fe-1/0/0.0, selected
                State: <Active Ext>
                Peer AS:     1
                Age: 45:46
                Task: BGP_1.192.168.197.141+179
                Announcement bits (2): 0-BGP.0.0.0.0+179 1-KRT
                AS path: 1 I
                Localpref: 100
                Router ID: 10.255.14.155
```

If the routes from the local CE router are not present in the VRF routing table, check that the CE router is advertising routes to the PE router. If static routing is used between the CE and PE routers, make sure the proper static routes are configured.

4. On a remote PE router, check that the routes from the local CE router are present in the bgp.l3vpn.0 routing table:

```
user@host> show route table bgp.l3vpn.0 extensive
10.255.14.175:3:10.255.14.155/32 (1 entry, 0 announced)
        *BGP    Preference: 170/-101
                Route Distinguisher: 10.255.14.175:3
                Source: 10.255.14.175
                Nexthop: 192.168.192.1 via fe-1/1/2.0, selected
                label-switched-path vpn07-vpn05
                Push 100004, Push 100005(top)
                State: <Active Int Ext>
                Local AS:    69 Peer AS:    69
                Age: 15:27      Metric2: 338
                Task: BGP_69.10.255.14.175+179
                AS path: 1 I
                Communities: target:69:100
                BGP next hop: 10.255.14.175
                Localpref: 100
                Router ID: 10.255.14.175
                Secondary tables: VPN-A.inet.0
```

The output of the `show route table bgp.l3vpn.0 extensive` command contains the following information specific to the VPN:

- In the prefix name (the first line of the output), the route distinguisher is added to the route prefix of the local CE router. Because the route distinguisher is unique within the Internet, the concatenation of the route distinguisher and IP prefix provides unique VPN-IP version 4 (IPv4) routing entries.

- The `Route Distinguisher` field lists the route distinguisher separately from the VPN-IPv4 address.

- The `label-switched-path` field shows the name of the label-switched path (LSP) used to carry the VPN traffic.

- The `Push` field shows both labels being carried in the VPN-IPv4 packet. The first label is the inner label, which is the VPN label that was assigned by the PE router. The second label is the outer label, which is an RSVP label.

- The `Communities` field lists the target community.

- The `Secondary tables` field lists other routing tables on this router into which this route has been installed.

If routes from the local CE router are not present in the bgp.l3vpn.0 routing table on the remote PE router, do the following:

- Check the VRF import filter on the remote PE router, which is configured in the `vrf-import` statement. (On the local PE router, you check the VRF export filter, which is configured with the `vrf-export` statement.)
- Check that there is an operational LSP or an LDP path between the PE routers. To do this, check that the IBGP next-hop addresses are in the inet.3 table.
- Check that the IBGP session between the PE routers is established and configured properly.
- Check for "hidden" routes, which usually means that routes were not labeled properly. To do this, use the `show route table bgp.l3vpn.0 hidden` command.
- Check that the inner label matches the inner VPN label that is assigned by the local PE router. To do this, use the `show route table mpls` command.

The following example shows the output of this command on the remote PE router. Here, the inner label is `100004`.

```
...
Push 100004, Push 10005 (top)
```

The following example shows the output of this command on the local PE router, which shows that the inner label of `100004` matches the inner label on the remote PE router:

```
...
100004             *[VPN/7] 06:56:25, metric 1
 > to 192.168.197.141 via fe-1/0/0.0, Pop
```

5. On the remote PE router, check that the routes from the local CE router are present in the VRF table (*routing-instance-name*.inet.0):

```
user@host> show route table routing-instance-name.inet.0 detail
10.255.14.155/32 (1 entry, 1 announced)
        *BGP    Preference: 170/-101
                Route Distinguisher: 10.255.14.175:3
                Source: 10.255.14.175
                Nexthop: 192.168.192.1 via fe-1/1/2.0, selected
                label-switched-path vpn07-vpn05
                Push 100004, Push 100005(top)
```

```
                State: <Secondary Active Int Ext>
                Local AS:    69 Peer AS:    69
                Age: 1:16:22    Metric2: 338
                Task: BGP_69.10.255.14.175+179
                Announcement bits (2): 1-KRT 2-VPN-A-RIP
                AS path: 1 I
                Communities: target:69:100
                BGP next hop: 10.255.14.175
                Localpref: 100
                Router ID: 10.255.14.175
                Primary Routing Table bgp.l3vpn.0
```

In this routing table, the route distinguisher is no longer prepended to the prefix. The last line, `Primary Routing Table`, lists the table from which this route was learned.

If the routes are not present in this routing table, but were present in the bgp.l3vpn.0 routing table on the local CE router, the routes might have not passed the VRF import policy on the remote PE router.

If a VPN-IPv4 route matches no `vrf-import` policy, the route does not show up in the bgp.l3vpn table at all and hence is not present in the VRF table. If this occurs, it might indicate that on the PE router, you have configured another `vrf-import` statement on another VPN (with a common target), and the routes show up in the bgp.l3vpn.0 table, but are imported into the wrong VPN.

6. On the remote CE router, check that the routes from the local CE router are present in the routing table (inet.0):

```
user@host> show route
```

If the routes are not present, check the routing protocol configuration between the remote PE and CE routers, and make sure that peers and adjacencies (or static routes) between the PE and CE routers are correct.

7. If you determine that routes originated from the local CE router are correct, check the routes originated from the remote CE router by repeating this procedure.

Example: Troubleshooting Layer 3 VPNs

IN THIS SECTION

This example shows how to use the `ping` command to check the accessibility of various routers in a VPN topology, and how to use the `traceroute` command to check the path that packets travel between the VPN routers.

Requirements

This example uses the following hardware and software components:

- M Series routers
- Junos OS Release 10.0R1 and later

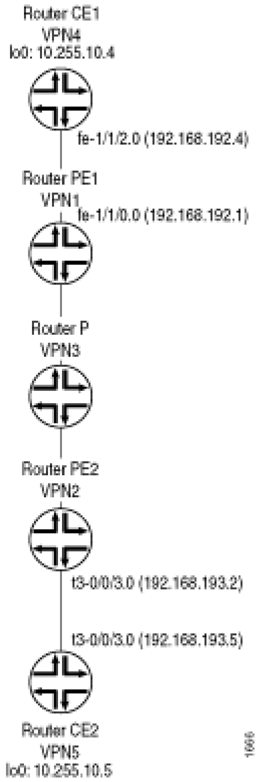
Overview

IN THIS SECTION

Topology

The topology shown in Figure 93 on page 1233 illustrates the network used in this example to demonstrate how to employ the ping and traceroute commands to test connectivity between the routers participating in a Layer 3 VPN.

Figure 93: Layer 3 VPN Topology for ping and traceroute Examples



Pinging the CE Router from Another CE Router

IN THIS SECTION

Procedure

Step-by-Step Procedure

The following describes how to use the ping and `traceroute` commands to troubleshoot Layer 3 VPN topologies. You can ping one CE router from the other by specifying the other CE router's loopback address as the IP address in the `ping` command. This `ping` command succeeds if the loopback addresses have been announced by the CE routers to their directly connected PE routers. The success of these `ping` commands also means that Router CE1 can ping any network devices beyond Router CE2, and vice versa. Figure 93 on page 1233 shows the topology referenced in the following steps:

1. Ping Router CE2 (VPN5) from Router CE1 (VPN4):

```
user@vpn4> ping 10.255.10.5 local 10.255.10.4 count 3
PING 10.255.10.5 (10.255.10.5): 56 data bytes
64 bytes from 10.255.10.5: icmp_seq=0 ttl=253 time=1.086 ms
64 bytes from 10.255.10.5: icmp_seq=1 ttl=253 time=0.998 ms
64 bytes from 10.255.10.5: icmp_seq=2 ttl=253 time=1.140 ms
--- 10.255.10.5 ping statistics ---
3 packets transmitted, 3 packets received, 0% packet loss
round-trip min/avg/max/stddev = 0.998/1.075/1.140/0.059 ms
```

2. To determine the path from Router CE1's loopback interface to Router CE2's loopback interface, use the `traceroute` command:

```
user@vpn4> traceroute 10.255.10.5 source 10.255.10.4
traceroute to 10.255.10.5 (10.255.10.5) from 10.255.10.4, 30 hops max, 40 byte packets
 1  vpn1-fe-110.isp-core.net (192.168.192.1)  0.680 ms  0.491 ms  0.456 ms
 2  vpn2-t3-001.isp-core.net (192.168.192.110)  0.857 ms  0.766 ms  0.754 ms
     MPLS Label=100005 CoS=0 TTL=1 S=1
 3  vpn5.isp-core.net (10.255.10.5)  0.825 ms  0.886 ms  0.732 ms
```

3. When you use the `traceroute` command to examine the path used by a Layer 3 VPN, the provider (P) routers in the service provider's network are not displayed. As shown above, the jump from Router VPN1 to Router VPN2 is displayed as a single hop. The P router (VPN3) shown in Figure 93 on page 1233 is not displayed.

4. Ping Router CE1 (VPN4) from Router CE2 (VPN5):

```
user@vpn5> ping 10.255.10.4 local 10.255.10.5 count 3
PING 10.255.10.4 (10.255.10.4): 56 data bytes
64 bytes from 10.255.10.4: icmp_seq=0 ttl=253 time=1.042 ms
64 bytes from 10.255.10.4: icmp_seq=1 ttl=253 time=0.998 ms
64 bytes from 10.255.10.4: icmp_seq=2 ttl=253 time=0.954 ms
--- 10.255.10.4 ping statistics ---
3 packets transmitted, 3 packets received, 0% packet loss
round-trip min/avg/max/stddev = 0.954/0.998/1.042/0.036 ms
```

5. To determine the path from Router CE2 to Router CE1, use the `traceroute` command:

```
user@vpn5> traceroute 10.255.10.4 source 10.255.10.5
traceroute to 10.255.10.4 (10.255.10.4) from 10.255.10.5, 30 hops max, 40 byte packets
 1  vpn-08-t3-003.isp-core.net (192.168.193.2)  0.686 ms  0.519 ms  0.548 ms
 2  vpn1-so-100.isp-core.net (192.168.192.100)  0.918 ms  0.869 ms  0.859 ms
     MPLS Label=100021 CoS=0 TTL=1 S=1
 3  vpn4.isp-core.net (10.255.10.4)  0.878 ms  0.760 ms  0.739 ms
```

Pinging the Remote PE and CE Routers from the Local CE Router

IN THIS SECTION

Procedure

Step-by-Step Procedure

From the local CE router, you can ping the VPN interfaces on the remote PE and CE routers, which are point-to-point interfaces. Figure 93 on page 1233 shows the topology referenced in the following examples:

1. Ping router CE2 from router CE1.

```
user@vpn4> ping 192.168.193.5 local 10.255.10.4 count 3
PING 192.168.193.5 (192.168.193.5): 56 data bytes
64 bytes from 192.168.193.5: icmp_seq=0 ttl=253 time=1.040 ms
64 bytes from 192.168.193.5: icmp_seq=1 ttl=253 time=0.891 ms
64 bytes from 192.168.193.5: icmp_seq=2 ttl=253 time=0.944 ms
--- 192.168.193.5 ping statistics ---
3 packets transmitted, 3 packets received, 0% packet loss
round-trip min/avg/max/stddev = 0.891/0.958/1.040/0.062 ms
```

2. To determine the path from Router CE1’s loopback interface to Router CE2’s directly connected interface, use the `traceroute` command:

```
user@vpn4> traceroute 192.168.193.5 source 10.255.10.4
traceroute to 192.168.193.5 (192.168.193.5) from 10.255.10.4, 30 hops max, 40 byte packets
 1  vpn1-fe-110.isp-core.net (192.168.192.1)  0.669 ms  0.508 ms  0.457 ms
 2  vpn2-t3-001.isp-core.net (192.168.192.110)  0.851 ms  0.769 ms  0.750 ms
     MPLS Label=100000 CoS=0 TTL=1 S=1
 3  vpn5-t3-003.isp-core.net (192.168.193.5)  0.829 ms  0.838 ms  0.731 ms
```

3. Ping Router PE2 (VPN2) from Router CE1 (VPN4). In this case, packets that originate at Router CE1 go to Router PE2, then to Router CE2, and back to Router PE2 before Router PE2 can respond to Internet Control Message Protocol (ICMP) requests. You can verify this by using the `traceroute` command.

```
user@vpn4> ping 192.168.193.2 local 10.255.10.4 count 3
PING 192.168.193.2 (192.168.193.2): 56 data bytes
64 bytes from 192.168.193.2: icmp_seq=0 ttl=254 time=1.080 ms
64 bytes from 192.168.193.2: icmp_seq=1 ttl=254 time=0.967 ms
64 bytes from 192.168.193.2: icmp_seq=2 ttl=254 time=0.983 ms
--- 192.168.193.2 ping statistics ---
3 packets transmitted, 3 packets received, 0% packet loss
round-trip min/avg/max/stddev = 0.967/1.010/1.080/0.050 ms
```

4. To determine the path from Router CE1 to Router PE2, use the `traceroute` command:

```
user@vpn4> traceroute 192.168.193.2 source 10.255.10.4
traceroute to 192.168.193.2 (192.168.193.2) from 10.255.10.4, 30 hops max, 40 byte packets
 1  vpn1-fe-110.isp-core.net (192.168.192.1)  0.690 ms  0.490 ms  0.458 ms
 2  vpn2-t3-003.isp-core.net (192.168.193.2)  0.846 ms  0.768 ms  0.749 ms
     MPLS Label=100000 CoS=0 TTL=1 S=1
 3  vpn5-t3-003.isp-core.net (192.168.193.5)  0.643 ms  0.703 ms  0.600 ms
 4  vpn-08-t3-003.isp-core.net (192.168.193.2)  0.810 ms  0.739 ms  0.729 ms
```

Pinging a CE Router from a Multiaccess Interface

IN THIS SECTION

Procedure

Step-by-Step Procedure

You cannot ping one CE router from the other if the VPN interface is a multiaccess interface, such as the `fe-1/1/2.0` interface on Router CE1. To ping Router CE1 from Router CE2, you must either include the `vrf-table-label` statement at the `[edit routing-instances` *`routing-instance-name`*`]` hierarchy level on Router PE1 or configure a static route on Router PE1 to the VPN interface of Router CE1. If you include the `vrf-table-label` statement to ping a router, you cannot configure a static route.

1. If you configure a static route on Router PE1 to the VPN interface of Router CE1, its next hop must point to Router CE1 (at the `[edit routing-instance` *`routing-instance-name`*`]` hierarchy level), and this route must be announced from Router PE1 to Router PE2 as shown in the following configuration:

```
[edit]
routing-instances {
    direct-multipoint {
        instance-type vrf;
        interface fe-1/1/0.0;
        route-distinguisher 69:1;
        vrf-import direct-import;
        vrf-export direct-export;
        routing-options {
            static {
                route 192.168.192.4/32 next-hop 192.168.192.4;
            }
        }
        protocols {
            bgp {
                group to-vpn4 {
                    peer-as 1;
                    neighbor 192.168.192.4;
                }
```

```
            }
        }
    }
    policy-options {
        policy-statement direct-export {
            term a {
                from protocol bgp;
                then {
                    community add direct-comm;
                    accept;
                }
            }
            term b {
                from {
                    protocol static;
                    route-filter 192.168.192.4/32 exact;
                }
                then {
                    community add direct-comm;
                    accept;
                }
            }
            term d {
                then reject;
            }
        }
    }
}
```

2. Now you can ping Router CE1 from Router CE2:

```
user@vpn5> ping 192.168.192.4 local 10.255.10.5 count 3
PING 192.168.192.4 (192.168.192.4): 56 data bytes
64 bytes from 192.168.192.4: icmp_seq=0 ttl=253 time=1.092 ms
64 bytes from 192.168.192.4: icmp_seq=1 ttl=253 time=1.019 ms
64 bytes from 192.168.192.4: icmp_seq=2 ttl=253 time=1.031 ms
--- 192.168.192.4 ping statistics ---
3 packets transmitted, 3 packets received, 0% packet loss
round-trip min/avg/max/stddev = 1.019/1.047/1.092/0.032 ms
```

3. To determine the path between these two interfaces, use the `traceroute` command:

```
user@vpn5> traceroute 192.168.192.4 source 10.255.10.5
traceroute to 192.168.192.4 (192.168.192.4) from 10.255.10.5, 30 hops max, 40 byte packets
 1  vpn-08-t3003.isp-core.net (192.168.193.2)  0.678 ms  0.549 ms  0.494 ms
 2  vpn1-so-100.isp-core.net (192.168.192.100)  0.873 ms  0.847 ms  0.844 ms
     MPLS Label=100021 CoS=0 TTL=1 S=1
 3  vpn4-fe-112.isp-core.net (192.168.192.4)  0.825 ms  0.743 ms  0.764 ms
```

Pinging the Directly Connected PE Routers from the CE Routers

IN THIS SECTION

Procedure

Step-by-Step Procedure

From the loopback interfaces on the CE routers, you can ping the VPN interface on the directly connected PE router. Figure 93 on page 1233 shows the topology referenced in this procedure:

1. From the loopback interface on Router CE1 (VPN4), ping the VPN interface, `fe-1/1/0.0`, on Router PE1:

```
user@vpn4> ping 192.168.192.1 local 10.255.10.4 count 3
PING 192.168.192.1 (192.168.192.1): 56 data bytes
64 bytes from 192.168.192.1: icmp_seq=0 ttl=255 time=0.885 ms
64 bytes from 192.168.192.1: icmp_seq=1 ttl=255 time=0.757 ms
64 bytes from 192.168.192.1: icmp_seq=2 ttl=255 time=0.734 ms
--- 192.168.192.1 ping statistics ---
3 packets transmitted, 3 packets received, 0% packet loss
round-trip min/avg/max/stddev = 0.734/0.792/0.885/0.066 ms
```

2. From the loopback interface on Router CE2 (VPN5), ping the VPN interface, `t3-0/0/3.0`, on Router PE2:

```
user@vpn5> ping 192.168.193.2 local 10.255.10.5 count 3
PING 192.168.193.2 (192.168.193.2): 56 data bytes
64 bytes from 192.168.193.2: icmp_seq=0 ttl=255 time=0.998 ms
64 bytes from 192.168.193.2: icmp_seq=1 ttl=255 time=0.834 ms
64 bytes from 192.168.193.2: icmp_seq=2 ttl=255 time=0.819 ms
--- 192.168.193.2 ping statistics ---
3 packets transmitted, 3 packets received, 0% packet loss
round-trip min/avg/max/stddev = 0.819/0.884/0.998/0.081 ms
```

3. From the loopback interface on Router CE2 (VPN5), ping the VPN interface, `t3-0/0/3.0`, on Router PE2:

```
user@vpn5> ping 192.168.193.2 local 10.255.10.5 count 3
PING 192.168.193.2 (192.168.193.2): 56 data bytes
64 bytes from 192.168.193.2: icmp_seq=0 ttl=255 time=0.998 ms
64 bytes from 192.168.193.2: icmp_seq=1 ttl=255 time=0.834 ms
64 bytes from 192.168.193.2: icmp_seq=2 ttl=255 time=0.819 ms
--- 192.168.193.2 ping statistics ---
3 packets transmitted, 3 packets received, 0% packet loss
round-trip min/avg/max/stddev = 0.819/0.884/0.998/0.081 ms
```

4. To determine the path from the loopback interface on Router CE2 to the VPN interfaces on Router PE2, use the `traceroute` command:

```
user@vpn5> traceroute 192.168.193.2 source 10.255.10.5
traceroute to 192.168.193.2 (192.168.193.2) from 10.255.10.5, 30 hops max, 40 byte packets
 1  vpn-08-t3003.isp-core.net (192.168.193.2)  0.852 ms  0.670 ms  0.656 ms
```

Pinging the Directly Connected CE Routers from the PE Routers

IN THIS SECTION

Procedure

Step-by-Step Procedure

From the VPN and loopback interfaces on the PE routers, you can ping the VPN interface on the directly connected CE router. Figure 93 on page 1233 shows the topology referenced in this procedure:

1. From the VPN interface on the PE router (router PE1), you can ping the VPN or loopback interface on the directly connected CE router (router CE1).

   From the VPN interface on Router PE1 (VPN1), ping the VPN interface, `fe-1/1/0.0`, on Router CE1:

   ```
   user@vpn1> ping 192.168.192.4 interface fe-1/1/0.0 local 192.168.192.1 count 3
   PING 192.168.192.4 (192.168.192.4): 56 data bytes
   64 bytes from 192.168.192.4: icmp_seq=0 ttl=255 time=0.866 ms
   64 bytes from 192.168.192.4: icmp_seq=1 ttl=255 time=0.728 ms
   64 bytes from 192.168.192.4: icmp_seq=2 ttl=255 time=0.753 ms
   --- 192.168.192.4 ping statistics ---
   3 packets transmitted, 3 packets received, 0% packet loss
   round-trip min/avg/max/stddev = 0.728/0.782/0.866/0.060 ms
   ```

2. From the VPN interface on Router PE1 (VPN1), ping the loopback interface, `10.255.10.4`, on Router CE1:

   ```
   user@vpn1> ping 10.255.10.4 interface fe-1/1/0.0 local 192.168.192.1 count 3
   PING 10.255.10.4 (10.255.10.4): 56 data bytes
   64 bytes from 10.255.10.4: icmp_seq=0 ttl=255 time=0.838 ms
   64 bytes from 10.255.10.4: icmp_seq=1 ttl=255 time=0.760 ms
   64 bytes from 10.255.10.4: icmp_seq=2 ttl=255 time=0.771 ms
   --- 10.255.10.4 ping statistics ---
   3 packets transmitted, 3 packets received, 0% packet loss
   round-trip min/avg/max/stddev = 0.760/0.790/0.838/0.034 ms
   ```

3. To determine the path from the VPN interface on Router PE1 to the VPN and loopback interfaces on Router CE1, respectively, use the following `traceroute` commands:

   ```
   user@vpn1> traceroute 10.255.10.4 interface fe-1/1/0.0 source 192.168.192.1
   traceroute to 10.255.10.4 (10.255.10.4) from 192.168.192.1, 30 hops max, 40 byte packets
    1  vpn4.isp-core.net (10.255.10.4)  0.842 ms  0.659 ms  0.621 ms
   user@vpn1>  traceroute 192.168.192.4 interface fe-1/1/0.0 source 192.168.192.1
   ```

```
traceroute to 192.168.192.4 (192.168.192.4) from 192.168.192.1, 30 hops max, 40 byte packets
 1  vpn4-fe-112.isp-core.net (192.168.192.4)  0.810 ms  0.662 ms  0.640 ms
```

4. From the VPN interface on Router PE2 (VPN2), ping the VPN interface, `t3-0/0/3.0`, on Router CE2:

```
user@vpn2> ping 192.168.193.5 interface t3-0/0/3.0 local 192.168.193.2 count 3
PING 192.168.193.5 (192.168.193.5): 56 data bytes
64 bytes from 192.168.193.5: icmp_seq=0 ttl=255 time=0.852 ms
64 bytes from 192.168.193.5: icmp_seq=1 ttl=255 time=0.909 ms
64 bytes from 192.168.193.5: icmp_seq=2 ttl=255 time=0.793 ms
--- 192.168.193.5 ping statistics ---
3 packets transmitted, 3 packets received, 0% packet loss
round-trip min/avg/max/stddev = 0.793/0.851/0.909/0.047 ms
```

5. From the VPN interface on Router PE2 (VPN2), ping the loopback interface, `10.255.10.5`, on Router CE2:

```
user@vpn2> ping 10.255.10.5 interface t3-0/0/3.0 local 192.168.193.2 count 3
PING 10.255.10.5 (10.255.10.5): 56 data bytes
64 bytes from 10.255.10.5: icmp_seq=0 ttl=255 time=0.914 ms
64 bytes from 10.255.10.5: icmp_seq=1 ttl=255 time=0.888 ms
64 bytes from 10.255.10.5: icmp_seq=2 ttl=255 time=1.066 ms
--- 10.255.10.5 ping statistics ---
3 packets transmitted, 3 packets received, 0% packet loss
round-trip min/avg/max/stddev = 0.888/0.956/1.066/0.079 ms
```

6. To determine the path from the VPN interface on Router PE2 to the VPN and loopback interfaces on Router CE2, respectively, use the following `traceroute` commands:

```
user@vpn2> traceroute 10.255.10.5 interface t3-0/0/3.0 source 192.168.193.2
traceroute to 10.255.10.5 (10.255.10.5) from 192.168.193.2, 30 hops max, 40 byte packets
 1  vpn5.isp-core.net (10.255.10.5)  1.009 ms  0.677 ms  0.633 ms
user@vpn2>  traceroute 192.168.193.5 interface t3-0/0/3.0 source 192.168.193.2
traceroute to 192.168.193.5 (192.168.193.5) from 192.168.193.2, 30 hops max, 40 byte packets
 1  vpn5-t3-003.isp-core.net (192.168.193.5)  0.974 ms  0.665 ms  0.619 ms
```

Pinging the Remote CE Router from the Local PE Router

Procedure

Step-by-Step Procedure

The following procedure is effective for Layer 3 VPNs only. To ping a remote CE router from a local PE router in a Layer 3 VPN, you need to configure the following interfaces:

1. Configure a logical unit for the loopback interface.

   To configure an additional logical unit on the loopback interface of the PE router, configure the `unit` statement at the `[edit interfaces lo0]` hierarchy level:

   ```
   [edit interfaces]
   lo0 {
       unit number {
           family inet {
               address address;
           }
       }
   }
   ```

2. Configure the loopback interface for the Layer 3 VPN routing instance on the local PE router. You can associate one logical loopback interface with each Layer 3 VPN routing instance, enabling you to ping a specific routing instance on a router.

   Specify the loopback interface you configured in Step "1" on page 1243 using the `interface` statement at the `[edit routing-instances routing-instance-name]` hierarchy level:

   ```
   [edit routing-instances routing-instance-name]
   interface interface-name;
   ```

   The *`interface-name`* is the logical unit on the loopback interface (for example, `lo0.1`).

3. From the VPN interface on PE router, you can now ping the logical unit on the loopback interface on the remote CE router:

   ```
   user@host> ping interface interface host
   ```

   Use *`interface`* to specify the new logical unit on the loopback interface (for example, `lo0.1`). For more information about how to use the `ping interface` command, see the *Junos Interfaces Command Reference*.

Troubleshooting Inconsistently Advertised Routes from Gigabit Ethernet Interfaces

IN THIS SECTION

Procedure

Step-by-Step Procedure

For direct routes on a LAN in a Layer 3 VPN, the Junos OS attempts to locate a CE router that can be designated as the next hop. If this cannot be done, advertised routes from Gigabit Ethernet interfaces are dropped.

In such instances:

1. Use the `static` statement at the `[edit routing-options]` or `[edit logical-systems` *`logical-system-name`* `routing-options]` hierarchy levels in the VRF routing instance to a CE router on the LAN subnet, configuring the CE router as the next hop. All traffic to directly destinations on this LAN will go to the CE router. You can add two static routes to two CE routers on the LAN for redundancy.

2. Configure the `vrf-table-label` statement at the `[edit routing-instances` *`routing-instance-name`*`]` hierarchy levels to map the inner label of a packet to a specific VRF routing table. This allows the examination of the encapsulated IP header to force IP lookups on the VRF routing instance for all traffic.

   > **NOTE**: The `vrf-table-label` statement is not available for every core-facing interface; for example, channelized interfaces are not supported. See "Filtering Packets in Layer 3 VPNs Based on IP Headers" on page 55 for information about support for the `vrf-table-label` statement over Ethernet and SONET/SDH interfaces.

Example: Diagnosing Networking Problems Related to Layer 3 VPNs by Disabling TTL Decrementing

IN THIS SECTION

This example shows how to disable TTL decrementing in a single VRF routing instance in a Layer 3 VPN scenario.

Requirements

Before you begin:

- Configure the router interfaces. See the *Network Interfaces Configuration Guide*.

Overview

IN THIS SECTION

To diagnose networking problems related to VPNs, it can be useful to disable normal time-to-live (TTL) decrementing. The IP header includes a TTL field that serves as a hop counter. At every routed hop, the TTL is decremented by one; if the TTL reaches zero before the packet reaches its destination, the packet is discarded and (optionally) an ICMP TTL exceeded message is sent to the source. MPLS labels also have a TTL field. MPLS routers copy the TTL of an IP packet when it enters a label-switched path (LSP). An IP packet with a TTL of 27 receives an MPLS label with a TTL of 27. Junos OS decrements the MPLS TTL of an MPLS-encapsulated packet in place of the IP TTL, at every label-switched hop. Because the MPLS TTL is copied (or propagated) from the IP TTL, a traceroute lists every hop in the path, be it routed or label-switched. When the packet exits the LSP, the decremented MPLS TTL is propagated back into the IP TTL field.

By default, TTL propagation is enabled. The global `no-propagate-ttl` statement disables TTL propagation at the router level and affects all RSVP-signalled or LDP-signalled LSPs. When a router acts as an ingress router for an LSP and the router configuration includes the `no-propagate-ttl` statement, the router pushes an MPLS header with a TTL value of 255, regardless of the IP packet TTL. When a router acts as the penultimate router, it pops the MPLS header without propagating the MPLS TTL into the IP packet. Thus the IP packet TTL value is preserved, regardless of the hop count of the LSP.

Instead of configuring TTL propagation behavior at the router level, you can configure the behavior for the routes in a VRF routing instance. This example shows how to disable TTL propagation for the routes in a single VRF routing instance instead of at the global router level.

The per-VRF configuration takes precedence over the global router configuration. If you disable TTL propagation on the router and explicitly enable TTL propagation for a single VRF routing instance, TTL propagation is in effect for that routing instance. To explicitly enable TTL propagation on a VRF routing instance, include the `vrf-propagate-ttl` statement in the routing instance.

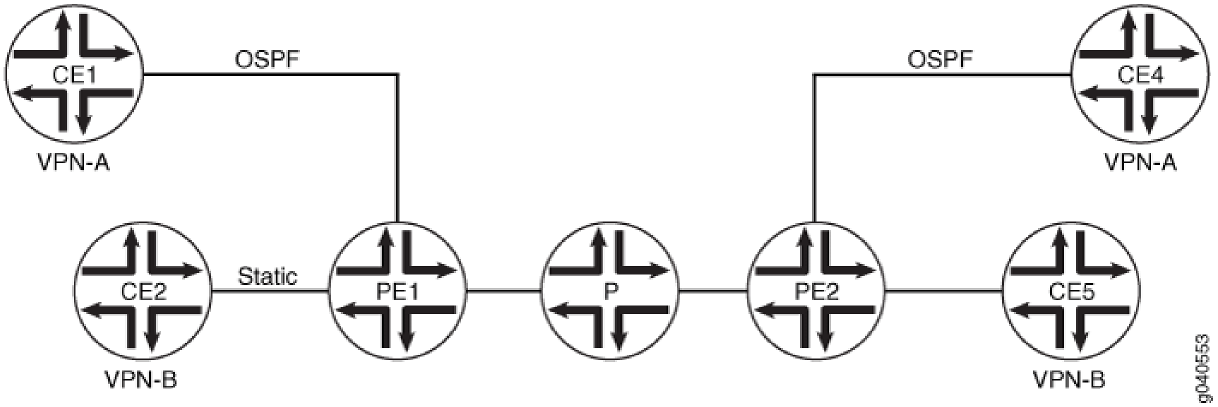
When you change the TTL propagation behavior, old next hops for VRF routes are deleted from the inet.3 routing table and new next hops are added.

You need only configure the `vrf-propagate-ttl` or `no-vrf-propagate-ttl` statement on the ingress routers.

Topology Diagram

Figure 94 on page 1246 shows the topology used in this example. Router PE1 and Router PE2 have two VPNs---VPN-A and VPN-B. Devices CE1 and CE4 belong to VPN-A. Devices CE2 and CE5 belong to VPN-B. In this example, Router PE1 has TTL propagation disabled on VPN-A but not on VPN-B. Packets received by PE1 on the interface connected to CE1 have TTL propagation disabled. This example shows the configuration on Router PE1. You do not need to include the `no-vrf-propagate-ttl` statement on the egress router (PE2).

Figure 94: Disabling TTL Propagation for a Single VPN



Configuration

IN THIS SECTION

Procedure

CLI Quick Configuration

To quickly disable TTL propagation in a VRF routing instance, copy the following commands and paste the commands into the CLI.

```
[edit]
set interfaces lo0 unit 0 family inet address 10.255.179.45/32 primary
set protocols mpls interface all
set protocols bgp group ibgp type internal
set protocols bgp group ibgp local-address 10.255.179.45
set protocols bgp group ibgp family inet-vpn unicast
set protocols bgp group ibgp neighbor 10.255.179.71
set protocols ospf area 0.0.0.0 interface fe-1/1/2.0
set protocols ospf area 0.0.0.0 interface fxp0.0 disable
set protocols ospf area 0.0.0.0 interface lo0.0
set protocols ldp interface all
set policy-options policy-statement VPN-A-export term a from protocol ospf
set policy-options policy-statement VPN-A-export term a from interface ge-1/2/0.0
set policy-options policy-statement VPN-A-export term a then community add VPN-A
set policy-options policy-statement VPN-A-export term a then accept
set policy-options policy-statement VPN-A-export term b then reject
set policy-options policy-statement VPN-A-import term a from protocol bgp
set policy-options policy-statement VPN-A-import term a from community VPN-A
set policy-options policy-statement VPN-A-import term a then accept
set policy-options policy-statement VPN-A-import term b then reject
set policy-options policy-statement VPN-B-export term a from protocol static
set policy-options policy-statement VPN-B-export term a then community add VPN-B
set policy-options policy-statement VPN-B-export term a then accept
set policy-options policy-statement VPN-B-export term b then reject
```

```
set policy-options policy-statement VPN-B-import term a from protocol bgp
set policy-options policy-statement VPN-B-import term a from community VPN-B
set policy-options policy-statement VPN-B-import term a then accept
set policy-options policy-statement VPN-B-import term b then reject
set policy-options policy-statement bgp-to-ospf from protocol bgp
set policy-options policy-statement bgp-to-ospf then accept
set policy-options community VPN-A members target:1:100
set policy-options community VPN-B members target:1:200
set routing-instances VPN-A instance-type vrf
set routing-instances VPN-A interface ge-1/2/0.0
set routing-instances VPN-A route-distinguisher 10.255.179.45:100
set routing-instances VPN-A interface ge-1/2/0.0
set routing-instances VPN-A no-vrf-propagate-ttl
set routing-instances VPN-A vrf-import VPN-A-import
set routing-instances VPN-A vrf-export VPN-A-export
set routing-instances VPN-A protocols ospf export bgp-to-ospf
set routing-instances VPN-A protocols ospf area 0.0.0.0 interface ge-1/2/0.0
set routing-instances VPN-B instance-type vrf
set routing-instances VPN-B interface so-0/1/0.0
set routing-instances VPN-B route-distinguisher 10.255.179.45:300
set routing-instances VPN-B vrf-import VPN-B-import
set routing-instances VPN-B vrf-export VPN-B-export
set routing-instances VPN-B routing-options static route 10.255.179.15/32 next-hop so-0/1/0.0
set routing-options autonomous-system 1
```

Step-by-Step Procedure

The following example requires you to navigate various levels in the configuration hierarchy. For information about navigating the CLI, see *Using the CLI Editor in Configuration Mode*.

To configure a flow map:

1. Configure the loopback interface.

   ```
   [edit]
   user@PE1# edit interfaces
   [edit interfaces]
   user@PE1# set lo0 unit 0 family inet address 10.255.179.45/32 primary
   user@PE1# exit
   ```

2. Configure the routing protocols.

The internal BGP neighbor address is the loopback interface address of Router PE2 in Figure 94 on page 1246.

```
[edit]
user@PE1# edit protocols
[edit protocols]
user@PE1# set mpls interface all
user@PE1# set bgp group ibgp type internal
user@PE1# set bgp group ibgp local-address 10.255.179.45
user@PE1# set bgp group ibgp family inet-vpn unicast
user@PE1# set bgp group ibgp neighbor 10.255.179.71
user@PE1# set ospf area 0.0.0.0 interface fe-1/1/2.0
user@PE1# set ospf area 0.0.0.0 interface fxp0.0 disable
user@PE1# set ospf area 0.0.0.0 interface lo0.0
user@PE1# set ldp interface all
user@PE1# exit
```

3. Configure routing policies for VPN-A and VPN-B.

```
[edit]
user@PE1# edit policy-options
[edit policy-options]
user@PE1# set policy-statement VPN-A-export term a from protocol ospf
user@PE1# set policy-statement VPN-A-export term a from interface ge-1/2/0.0
user@PE1# set policy-statement VPN-A-export term a then community add VPN-A
user@PE1# set policy-statement VPN-A-export term a then accept
user@PE1# set policy-statement VPN-A-export term b then reject
user@PE1# set policy-statement VPN-A-import term a from protocol bgp
user@PE1# set policy-statement VPN-A-import term a from community VPN-A
user@PE1# set policy-statement VPN-A-import term a then accept
user@PE1# set policy-statement VPN-A-import term b then reject
user@PE1# set policy-statement VPN-B-export term a from protocol static
user@PE1# set policy-statement VPN-B-export term a then community add VPN-B
user@PE1# set policy-statement VPN-B-export term a then accept
user@PE1# set policy-statement VPN-B-export term b then reject
user@PE1# set policy-statement VPN-B-import term a from protocol bgp
user@PE1# set policy-statement VPN-B-import term a from community VPN-B
user@PE1# set policy-statement VPN-B-import term a then accept
user@PE1# set policy-statement VPN-B-import term b then reject
user@PE1# set policy-statement bgp-to-ospf from protocol bgp
user@PE1# set policy-statement bgp-to-ospf then accept
```

```
user@PE1# set community VPN-A members target:1:100
user@PE1# set community VPN-B members target:1:200
user@PE1# exit
```

4. Configure the VPN-A and VPN-B routing instances, including the `no-vrf-propagate-ttl` statement in VPN-A.

```
[edit]
user@PE1# edit routing-instances
[edit routing-instances]
user@PE1# set VPN-A instance-type vrf
user@PE1# set VPN-A interface ge-1/2/0.0
user@PE1# set VPN-A route-distinguisher 10.255.179.45:100
user@PE1# set VPN-A interface ge-1/2/0.0
user@PE1# set VPN-A no-vrf-propagate-ttl
user@PE1# set VPN-A vrf-import VPN-A-import
user@PE1# set VPN-A vrf-export VPN-A-export
user@PE1# set VPN-A protocols ospf export bgp-to-ospf
user@PE1# set VPN-A protocols ospf area 0.0.0.0 interface ge-1/2/0.0
user@PE1# set VPN-B instance-type vrf
user@PE1# set VPN-B interface so-0/1/0.0
user@PE1# set VPN-B route-distinguisher 10.255.179.45:300
user@PE1# set VPN-B vrf-import VPN-B-import
user@PE1# set VPN-B vrf-export VPN-B-export
user@PE1# set VPN-B routing-options static route 10.255.179.15/32 next-hop so-0/1/0.0
user@PE1# exit
```

5. Define the local autonomous system.

```
[edit]
user@PE1# edit routing-options
[edit routing-options]
user@PE1# set autonomous-system 1
user@PE1# exit
```

6. If you are done configuring the device, commit the configuration.

```
[edit]
user@PE1# commit
```

Results

Confirm your configuration by entering the `show interfaces`, `show policy-options`, `show protocols`, `show routing-instances`, and `show routing-options` commands.

```
user@PE1# show interfaces
lo0 {
    unit 0 {
        family inet {
            address 10.255.179.45/32 {
                primary;
            }
        }
    }
}
```

```
user@PE1# show policy-options
policy-statement VPN-A-export {
    term a {
        from {
            protocol ospf;
            interface ge-1/2/0.0;
        }
        then {
            community add VPN-A;
            accept;
        }
    }
    term b {
        then reject;
    }
}
policy-statement VPN-A-import {
    term a {
        from {
            protocol bgp;
            community VPN-A;
        }
        then accept;
    }
    term b {
```

```
        then reject;
    }
}
policy-statement VPN-B-export {
    term a {
        from protocol static;
        then {
            community add VPN-B;
            accept;
        }
    }
    term b {
        then reject;
    }
}
policy-statement VPN-B-import {
    term a {
        from {
            protocol bgp;
            community VPN-B;
        }
        then accept;
    }
    term b {
        then reject;
    }
}
policy-statement bgp-to-ospf {
    from protocol bgp;
    then accept;
}
community VPN-A members target:1:100;
community VPN-B members target:1:200;
```

```
user@PE1# show protocols
mpls {
    interface all;
}
bgp {
    group ibgp {
        type internal;
```

```
        local-address 10.255.179.45;
        family inet-vpn {
            unicast;
        }
        neighbor 10.255.179.71;
    }
}
ospf {
    area 0.0.0.0 {
        interface fe-1/1/2.0;
        interface fxp0.0 {
            disable;
        }
        interface lo0.0;
    }
}
ldp {
    interface all;
}
```

```
user@PE1# show routing-instances
VPN-A {
    instance-type vrf;
    interface ge-1/2/0.0;
    no-vrf-propagate-ttl;
    route-distinguisher 10.255.179.45:100;
    vrf-import VPN-A-import;
    vrf-export VPN-A-export;
    protocols {
        ospf {
            export bgp-to-ospf;
            area 0.0.0.0 {
                interface ge-1/2/0.0;
            }
        }
    }
}
VPN-B {
    instance-type vrf;
    interface so-0/1/0.0;
    route-distinguisher 10.255.179.45:300;
```

```
    vrf-import VPN-B-import;
    vrf-export VPN-B-export;
    routing-options {
        static {
            route 10.255.179.15/32 next-hop so-0/1/0.0;
        }
    }
}
```

```
user@PE1# show routing-options
autonomous-system 1;
```

Verification

To verify the operation, run the following commands:

- See the `TTL Action` field in the output of the `show route extensive table VPN-A` command.
- See the `TTL Action` field in the output of the `show route extensive table VPN-B` command.
- On Device CE1, run the `traceroute` command to Device CE4's loopback address.
- On Device CE4, run the `traceroute` command to Device CE1's loopback address.

SEE ALSO

Disabling Normal TTL Decrementing

2

PART

Configuration Statements and Operational Commands

Junos CLI Reference Overview

We've consolidated all Junos CLI commands and configuration statements in one place. Learn about the syntax and options that make up the statements and commands and understand the contexts in which you'll use these CLI elements in your network configurations and operations.

- Junos CLI Reference

Click the links to access Junos OS and Junos OS Evolved configuration statement and command summary topics.

- Configuration Statements
- Operational Commands