JUNIPER NETWORKS

Junos® OS

VPNs Configuration Guide

Release

11.1

Published: 2011-02-16

Juniper Networks, Inc.
1194 North Mathilda Avenue
Sunnyvale, California 94089
USA
408-745-2000
www.juniper.net

Junos® OS VPNs Configuration Guide
Release 11.1

Revision History
January 2011—R1 Junos OS 11.1

The information in this document is current as of the date listed in the revision history.

YEAR 2000 NOTICE

Juniper Networks hardware and software products are Year 2000 compliant. The 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

READ THIS END USER LICENSE AGREEMENT ("AGREEMENT") BEFORE DOWNLOADING, INSTALLING, OR USING THE SOFTWARE. BY DOWNLOADING, INSTALLING, OR USING THE SOFTWARE OR OTHERWISE EXPRESSING YOUR AGREEMENT TO THE TERMS CONTAINED HEREIN, YOU (AS CUSTOMER OR IF YOU ARE NOT THE CUSTOMER, AS A REPRESENTATIVE/AGENT AUTHORIZED TO BIND THE CUSTOMER) CONSENT TO BE BOUND BY THIS AGREEMENT. IF YOU DO NOT OR CANNOT AGREE TO THE TERMS CONTAINED HEREIN, THEN (A) DO NOT DOWNLOAD, INSTALL, OR USE THE SOFTWARE, AND (B) YOU MAY CONTACT JUNIPER NETWORKS REGARDING LICENSE TERMS.

1. **The Parties.** The parties to this Agreement are (i) Juniper Networks, Inc. (if the Customer's principal office is located in the Americas) or Juniper Networks (Cayman) Limited (if the Customer's principal office is located outside the Americas) (such applicable entity being referred to herein as "Juniper"), and (ii) the person or organization that originally purchased from Juniper or an authorized Juniper reseller the applicable license(s) for use of the Software ("Customer") (collectively, the "Parties").

2. **The Software.** In this Agreement, "Software" means the program modules and features of the Juniper or Juniper-supplied software, for which Customer has paid the applicable license or support fees to Juniper or an authorized Juniper reseller, or which was embedded by Juniper in equipment which Customer purchased from Juniper or an authorized Juniper reseller. "Software" also includes updates, upgrades and new releases of such software. "Embedded Software" means Software which Juniper has embedded in or loaded onto the Juniper equipment and any updates, upgrades, additions or replacements which are subsequently embedded in or loaded onto the equipment.

3. **License Grant.** Subject to payment of the applicable fees and the limitations and restrictions set forth herein, Juniper grants to Customer a non-exclusive and non-transferable license, without right to sublicense, to use the Software, in executable form only, subject to the following use restrictions:

a. Customer shall use Embedded Software solely as embedded in, and for execution on, Juniper equipment originally purchased by Customer from Juniper or an authorized Juniper reseller.

b. Customer shall use the Software on a single hardware chassis having a single processing unit, or as many chassis or processing units for which Customer has paid the applicable license fees; provided, however, with respect to the Steel-Belted Radius or Odyssey Access Client software only, Customer shall use such Software on a single computer containing a single physical random access memory space and containing any number of processors. Use of the Steel-Belted Radius or IMS AAA software on multiple computers or virtual machines (e.g., Solaris zones) requires multiple licenses, regardless of whether such computers or virtualizations are physically contained on a single chassis.

c. Product purchase documents, paper or electronic user documentation, and/or the particular licenses purchased by Customer may specify limits to Customer's use of the Software. Such limits may restrict use to a maximum number of seats, registered endpoints, concurrent users, sessions, calls, connections, subscribers, clusters, nodes, realms, devices, links, ports or transactions, or require the purchase of separate licenses to use particular features, functionalities, services, applications, operations, or capabilities, or provide throughput, performance, configuration, bandwidth, interface, processing, temporal, or geographical limits. In addition, such limits may restrict the use of the Software to managing certain kinds of networks or require the Software to be used only in conjunction with other specific Software. Customer's use of the Software shall be subject to all such limitations and purchase of all applicable licenses.

d. For any trial copy of the Software, Customer's right to use the Software expires 30 days after download, installation or use of the Software. Customer may operate the Software after the 30-day trial period only if Customer pays for a license to do so. Customer may not extend or create an additional trial period by re-installing the Software after the 30-day trial period.

e. The Global Enterprise Edition of the Steel-Belted Radius software may be used by Customer only to manage access to Customer's enterprise network. Specifically, service provider customers are expressly prohibited from using the Global Enterprise Edition of the Steel-Belted Radius software to support any commercial network access services.

The foregoing license is not transferable or assignable by Customer. No license is granted herein to any user who did not originally purchase the applicable license(s) for the Software from Juniper or an authorized Juniper reseller.

4. **Use Prohibitions.** Notwithstanding the foregoing, the license provided herein does not permit the Customer to, and Customer agrees not to and shall not: (a) modify, unbundle, reverse engineer, or create derivative works based on the Software; (b) make unauthorized copies of the Software (except as necessary for backup purposes); (c) rent, sell, transfer, or grant any rights in and to any copy of the Software, in any form, to any third party; (d) remove any proprietary notices, labels, or marks on or in any copy of the Software or any product in which the Software is embedded; (e) distribute any copy of the Software to any third party, including as may be embedded in Juniper equipment sold in the secondhand market; (f) use any 'locked' or key-restricted feature, function, service, application, operation, or capability without first purchasing the applicable license(s) and obtaining a valid key from Juniper, even if such feature, function, service, application, operation, or capability is enabled without a key; (g) distribute any key for the Software provided by Juniper to any third party; (h) use the

Software in any manner that extends or is broader than the uses purchased by Customer from Juniper or an authorized Juniper reseller; (i) use Embedded Software on non-Juniper equipment; (j) use Embedded Software (or make it available for use) on Juniper equipment that the Customer did not originally purchase from Juniper or an authorized Juniper reseller; (k) disclose the results of testing or benchmarking of the Software to any third party without the prior written consent of Juniper; or (l) use the Software in any manner other than as expressly provided herein.

5. **Audit.** Customer shall maintain accurate records as necessary to verify compliance with this Agreement. Upon request by Juniper, Customer shall furnish such records to Juniper and certify its compliance with this Agreement.

6. **Confidentiality.** The Parties agree that aspects of the Software and associated documentation are the confidential property of Juniper. As such, Customer shall exercise all reasonable commercial efforts to maintain the Software and associated documentation in confidence, which at a minimum includes restricting access to the Software to Customer employees and contractors having a need to use the Software for Customer's internal business purposes.

7. **Ownership.** Juniper and Juniper's licensors, respectively, retain ownership of all right, title, and interest (including copyright) in and to the Software, associated documentation, and all copies of the Software. Nothing in this Agreement constitutes a transfer or conveyance of any right, title, or interest in the Software or associated documentation, or a sale of the Software, associated documentation, or copies of the Software.

8. **Warranty, Limitation of Liability, Disclaimer of Warranty.** The warranty applicable to the Software shall be as set forth in the warranty statement that accompanies the Software (the "Warranty Statement"). Nothing in this Agreement shall give rise to any obligation to support the Software. Support services may be purchased separately. Any such support shall be governed by a separate, written support services agreement. TO THE MAXIMUM EXTENT PERMITTED BY LAW, JUNIPER SHALL NOT BE LIABLE FOR ANY LOST PROFITS, LOSS OF DATA, OR COSTS OR PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES, OR FOR ANY SPECIAL, INDIRECT, OR CONSEQUENTIAL DAMAGES ARISING OUT OF THIS AGREEMENT, THE SOFTWARE, OR ANY JUNIPER OR JUNIPER-SUPPLIED SOFTWARE. IN NO EVENT SHALL JUNIPER BE LIABLE FOR DAMAGES ARISING FROM UNAUTHORIZED OR IMPROPER USE OF ANY JUNIPER OR JUNIPER-SUPPLIED SOFTWARE. EXCEPT AS EXPRESSLY PROVIDED IN THE WARRANTY STATEMENT TO THE EXTENT PERMITTED BY LAW, JUNIPER DISCLAIMS ANY AND ALL WARRANTIES IN AND TO THE SOFTWARE (WHETHER EXPRESS, IMPLIED, STATUTORY, OR OTHERWISE), INCLUDING ANY IMPLIED WARRANTY OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, OR NONINFRINGEMENT. IN NO EVENT DOES JUNIPER WARRANT THAT THE SOFTWARE, OR ANY EQUIPMENT OR NETWORK RUNNING THE SOFTWARE, WILL OPERATE WITHOUT ERROR OR INTERRUPTION, OR WILL BE FREE OF VULNERABILITY TO INTRUSION OR ATTACK. In no event shall Juniper's or its suppliers' or licensors' liability to Customer, whether in contract, tort (including negligence), breach of warranty, or otherwise, exceed the price paid by Customer for the Software that gave rise to the claim, or if the Software is embedded in another Juniper product, the price paid by Customer for such other product. Customer acknowledges and agrees that Juniper has set its prices and entered into this Agreement in reliance upon the disclaimers of warranty and the limitations of liability set forth herein, that the same reflect an allocation of risk between the Parties (including the risk that a contract remedy may fail of its essential purpose and cause consequential loss), and that the same form an essential basis of the bargain between the Parties.

9. **Termination.** Any breach of this Agreement or failure by Customer to pay any applicable fees due shall result in automatic termination of the license granted herein. Upon such termination, Customer shall destroy or return to Juniper all copies of the Software and related documentation in Customer's possession or control.

10. **Taxes.** All license fees payable under this agreement are exclusive of tax. Customer shall be responsible for paying Taxes arising from the purchase of the license, or importation or use of the Software. If applicable, valid exemption documentation for each taxing jurisdiction shall be provided to Juniper prior to invoicing, and Customer shall promptly notify Juniper if their exemption is revoked or modified. All payments made by Customer shall be net of any applicable withholding tax. Customer will provide reasonable assistance to Juniper in connection with such withholding taxes by promptly: providing Juniper with valid tax receipts and other required documentation showing Customer's payment of any withholding taxes; completing appropriate applications that would reduce the amount of withholding tax to be paid; and notifying and assisting Juniper in any audit or tax proceeding related to transactions hereunder. Customer shall comply with all applicable tax laws and regulations, and Customer will promptly pay or reimburse Juniper for all costs and damages related to any liability incurred by Juniper as a result of Customer's non-compliance or delay with its responsibilities herein. Customer's obligations under this Section shall survive termination or expiration of this Agreement.

11. **Export.** Customer agrees to comply with all applicable export laws and restrictions and regulations of any United States and any applicable foreign agency or authority, and not to export or re-export the Software or any direct product thereof in violation of any such restrictions, laws or regulations, or without all necessary approvals. Customer shall be liable for any such violations. The version of the Software supplied to Customer may contain encryption or other capabilities restricting Customer's ability to export the Software without an export license.

12. **Commercial Computer Software.** The Software is "commercial computer software" and is provided with restricted rights. Use, duplication, or disclosure by the United States government is subject to restrictions set forth in this Agreement and as provided in DFARS 227.7201 through 227.7202-4, FAR 12.212, FAR 27.405(b)(2), FAR 52.227-19, or FAR 52.227-14(ALT III) as applicable.

13. **Interface Information.** To the extent required by applicable law, and at Customer's written request, Juniper shall provide Customer with the interface information needed to achieve interoperability between the Software and another independently created program, on payment of applicable fee, if any. Customer shall observe strict obligations of confidentiality with respect to such information and shall use such information in compliance with any applicable terms and conditions upon which Juniper makes such information available.

14. **Third Party Software.** Any licensor of Juniper whose software is embedded in the Software and any supplier of Juniper whose products or technology are embedded in (or services are accessed by) the Software shall be a third party beneficiary with respect to this Agreement, and such licensor or vendor shall have the right to enforce this Agreement in its own name as if it were Juniper. In addition, certain third party software may be provided with the Software and is subject to the accompanying license(s), if any, of its respective owner(s). To the extent portions of the Software are distributed under and subject to open source licenses obligating Juniper to make the source code for such portions publicly available (such as the GNU General Public License ("GPL") or the GNU Library General Public License ("LGPL")), Juniper will make such source code portions (including Juniper modifications, as appropriate) available upon request for a period of up to three years from the date of distribution. Such request can be made in writing to Juniper Networks, Inc., 1194 N. Mathilda Ave., Sunnyvale, CA 94089, ATTN: General Counsel. You may obtain a copy of the GPL at http://www.gnu.org/licenses/gpl.html, and a copy of the LGPL at http://www.gnu.org/licenses/lgpl.html .

15. **Miscellaneous.** This Agreement shall be governed by the laws of the State of California without reference to its conflicts of laws principles. The provisions of the U.N. Convention for the International Sale of Goods shall not apply to this Agreement. For any disputes arising under this Agreement, the Parties hereby consent to the personal and exclusive jurisdiction of, and venue in, the state and federal courts within Santa Clara County, California. This Agreement constitutes the entire and sole agreement between Juniper and the Customer with respect to the Software, and supersedes all prior and contemporaneous agreements relating to the Software, whether oral or written (including any inconsistent terms contained in a purchase order), except that the terms of a separate written agreement executed by an authorized Juniper representative and Customer shall govern to the extent such terms are inconsistent or conflict with terms contained herein. No modification to this Agreement nor any waiver of any rights hereunder shall be effective unless expressly assented to in writing by the party to be charged. If any portion of this Agreement is held invalid, the Parties agree that such invalidity shall not affect the validity of the remainder of this Agreement. This Agreement and associated documentation has been written in the English language, and the Parties agree that the English version will govern. (For Canada: Les parties aux présentés confirment leur volonté que cette convention de même que tous les documents y compris tout avis qui s'y rattaché, soient redigés en langue anglaise. (Translation: The parties confirm that this Agreement and all related documentation is and will be in the English language)).

Abbreviated Table of Contents

Table of Contents

Part 7 Layer 2 Circuits

List of Figures

List of Tables

About This Guide

This preface provides the following guidelines for using the *Junos® OS VPNs Configuration Guide*:

Junos OS Documentation and Release Notes

For a list of related Junos OS documentation, see http://www.juniper.net/techpubs/software/junos/ .

If the information in the latest release notes differs from the information in the documentation, follow the *Junos OS Release Notes.*

To obtain the most current version of all Juniper Networks® technical documentation, see the product documentation page on the Juniper Networks website at http://www.juniper.net/techpubs/ .

Juniper Networks supports a technical book program to publish books by Juniper Networks engineers and subject matter experts with book publishers around the world. These books go beyond the technical documentation to explore the nuances of network architecture, deployment, and administration using the Junos operating system (Junos OS) and Juniper Networks devices. In addition, the Juniper Networks Technical Library, published in conjunction with O'Reilly Media, explores improving network security, reliability, and availability using Junos OS configuration techniques. All the books are for sale at technical bookstores and book outlets around the world. The current list can be viewed at http://www.juniper.net/books .

Objectives

This guide provides an overview of and describes how to configure the Juniper Networks Junos OS virtual private network (VPN) functions, virtual private LAN service (VPLS) functions, and Layer 2 circuit functions.

NOTE: For additional information about the Junos OS—either corrections to or information that might have been omitted from this guide—see the software release notes at http://www.juniper.net/ .

Audience

This guide is designed for network administrators who are configuring and monitoring a Juniper Networks M Series, MX Series, T Series, EX Series, or J Series router or switch.

To use this guide, you need a broad understanding of networks in general, the Internet in particular, networking principles, and network configuration. You must also be familiar with one or more of the following Internet routing protocols:

- Border Gateway Protocol (BGP)
- Distance Vector Multicast Routing Protocol (DVMRP)
- Intermediate System-to-Intermediate System (IS-IS)
- Internet Control Message Protocol (ICMP) router discovery
- Internet Group Management Protocol (IGMP)
- Multiprotocol Label Switching (MPLS)
- Open Shortest Path First (OSPF)
- Protocol-Independent Multicast (PIM)
- Resource Reservation Protocol (RSVP)
- Routing Information Protocol (RIP)
- Simple Network Management Protocol (SNMP)

Personnel operating the equipment must be trained and competent; must not conduct themselves in a careless, willfully negligent, or hostile manner; and must abide by the instructions provided by the documentation.

Supported Platforms

For the features described in this manual, the Junos OS currently supports the following platforms:

- J Series
- M Series

- MX Series
- T Series
- EX Series

Using the Indexes

This reference contains two indexes: a standard index with topic entries, and an index of commands.

Using the Examples in This Manual

If you want to use the examples in this manual, you can use the **load merge** or the **load merge relative** command. These commands cause the software to merge the incoming configuration into the current candidate configuration. If the example configuration contains the top level of the hierarchy (or multiple hierarchies), the example is a *full example*. In this case, use the **load merge** command.

If the example configuration does not start at the top level of the hierarchy, the example is a *snippet*. In this case, use the **load merge relative** command. These procedures are described in the following sections.

Merging a Full Example

To merge a full example, follow these steps:

1. From the HTML or PDF version of the manual, copy a configuration example into a text file, save the file with a name, and copy the file to a directory on your routing platform.

   For example, copy the following configuration to a file and name the file **ex-script.conf**. Copy the **ex-script.conf** file to the **/var/tmp** directory on your routing platform.

   ```
   system {
     scripts {
       commit {
         file ex-script.xsl;
       }
     }
   }
   interfaces {
     fxp0 {
       disable;
       unit 0 {
         family inet {
           address 10.0.0.1/24;
         }
       }
     }
   }
   ```

2. Merge the contents of the file into your routing platform configuration by issuing the **load merge** configuration mode command:

```
[edit]
user@host# load merge /var/tmp/ex-script.conf
load complete
```

Merging a Snippet

To merge a snippet, follow these steps:

1. From the HTML or PDF version of the manual, copy a configuration snippet into a text file, save the file with a name, and copy the file to a directory on your routing platform.

   For example, copy the following snippet to a file and name the file **ex-script-snippet.conf**. Copy the **ex-script-snippet.conf** file to the **/var/tmp** directory on your routing platform.

   ```
   commit {
     file ex-script-snippet.xsl; }
   ```

2. Move to the hierarchy level that is relevant for this snippet by issuing the following configuration mode command:

   ```
   [edit]
   user@host# edit system scripts
   [edit system scripts]
   ```

3. Merge the contents of the file into your routing platform configuration by issuing the **load merge relative** configuration mode command:

   ```
   [edit system scripts]
   user@host# load merge relative /var/tmp/ex-script-snippet.conf
   load complete
   ```

For more information about the **load** command, see the *Junos OS CLI User Guide*.

Documentation Conventions

Table 1 on page xxxviii defines notice icons used in this guide.

Table 1: Notice Icons

Icon	Meaning	Description
	Informational note	Indicates important features or instructions.
	Caution	Indicates a situation that might result in loss of data or hardware damage.
	Warning	Alerts you to the risk of personal injury or death.
	Laser warning	Alerts you to the risk of personal injury from a laser.

Table 2 on page xxxix defines the text and syntax conventions used in this guide.

Table 2: Text and Syntax Conventions

Convention	Description	Examples
Bold text like this	Represents text that you type.	To enter configuration mode, type the **configure** command: user@host> **configure**
`Fixed-width text like this`	Represents output that appears on the terminal screen.	`user@host>` **show chassis alarms** `No alarms currently active`
Italic text like this	• Introduces important new terms. • Identifies book names. • Identifies RFC and Internet draft titles.	• A policy *term* is a named structure that defines match conditions and actions. • *Junos OS System Basics Configuration Guide* • RFC 1997, *BGP Communities Attribute*
Italic text like this	Represents variables (options for which you substitute a value) in commands or configuration statements.	Configure the machine's domain name: [edit] root@# **set system domain-name** *domain-name*
Text like this	Represents names of configuration statements, commands, files, and directories; interface names; configuration hierarchy levels; or labels on routing platform components.	• To configure a stub area, include the **stub** statement at the **[edit protocols ospf area area-id]** hierarchy level. • The console port is labeled **CONSOLE**.
< > (angle brackets)	Enclose optional keywords or variables.	**stub** <**default-metric** *metric*>**;**
| (pipe symbol)	Indicates a choice between the mutually exclusive keywords or variables on either side of the symbol. The set of choices is often enclosed in parentheses for clarity.	**broadcast | multicast** (***string1*** **|** ***string2*** **|** ***string3***)
# (pound sign)	Indicates a comment specified on the same line as the configuration statement to which it applies.	**rsvp { # Required for dynamic MPLS only**
[ ] (square brackets)	Enclose a variable for which you can substitute one or more values.	**community name members [** ***community-ids*** **]**

Indention and braces ({ })	Identify a level in the configuration hierarchy.	[edit]
		routing-options {
		  static {
		  route default {
		  nexthop *address*;
		  retain;
		  }
		  }
		}

; (semicolon)	Identifies a leaf statement at a configuration hierarchy level.	

Table 2: Text and Syntax Conventions *(continued)*

Convention	Description	Examples
J-Web GUI Conventions		
Bold text like this	Represents J-Web graphical user interface (GUI) items you click or select.	• In the Logical Interfaces box, select **All Interfaces**. • To cancel the configuration, click **Cancel**.
> (bold right angle bracket)	Separates levels in a hierarchy of J-Web selections.	In the configuration editor hierarchy, select **Protocols>Ospf**.

Documentation Feedback

We encourage you to provide feedback, comments, and suggestions so that we can improve the documentation. You can send your comments to techpubs-comments@juniper.net, or fill out the documentation feedback form at https://www.juniper.net/cgi-bin/docbugreport/ . If you are using e-mail, be sure to include the following information with your comments:

- Document or topic name
- URL or page number
- Software release version (if applicable)

Requesting Technical Support

Technical product support is available through the Juniper Networks Technical Assistance Center (JTAC). If you are a customer with an active J-Care or JNASC support contract, or are covered under warranty, and need postsales technical support, you can access our tools and resources online or open a case with JTAC.

- JTAC policies—For a complete understanding of our JTAC procedures and policies, review the JTAC User Guide located at http://www.juniper.net/us/en/local/pdf/resource-guides/7100059-en.pdf .
- Product warranties—For product warranty information, visit http://www.juniper.net/support/warranty/ .
- JTAC Hours of Operation —The JTAC centers have resources available 24 hours a day, 7 days a week, 365 days a year.

Self-Help Online Tools and Resources

For quick and easy problem resolution, Juniper Networks has designed an online self-service portal called the Customer Support Center (CSC) that provides you with the following features:

- Find CSC offerings: http://www.juniper.net/customers/support/
- Find product documentation: http://www.juniper.net/techpubs/
- Find solutions and answer questions using our Knowledge Base: http://kb.juniper.net/
- Download the latest versions of software and review release notes: http://www.juniper.net/customers/csc/software/
- Search technical bulletins for relevant hardware and software notifications: https://www.juniper.net/alerts/
- Join and participate in the Juniper Networks Community Forum: http://www.juniper.net/company/communities/
- Open a case online in the CSC Case Management tool: http://www.juniper.net/cm/

To verify service entitlement by product serial number, use our Serial Number Entitlement (SNE) Tool: https://tools.juniper.net/SerialNumberEntitlementSearch/

Opening a Case with JTAC

You can open a case with JTAC on the Web or by telephone.

- Use the Case Management tool in the CSC at http://www.juniper.net/cm/ .
- Call 1-888-314-JTAC (1-888-314-5822 toll-free in the USA, Canada, and Mexico).

For international or direct-dial options in countries without toll-free numbers, visit us at http://www.juniper.net/support/requesting-support.html

PART 1

VPN Overview

CHAPTER 1

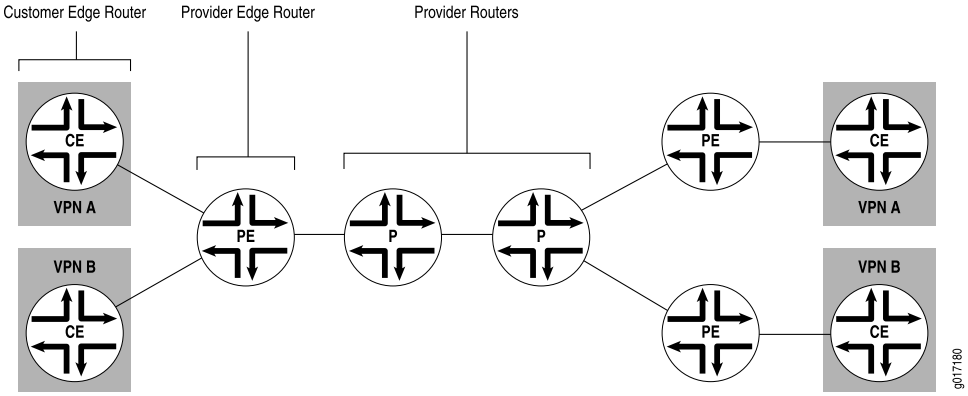
VPN Overview

This chapter discusses the following topics that provide background information about VPNs:

Routers in a VPN

Figure 1 on page 3 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 1: Routers in a VPN



VPN Terminology

C

Customer edge (CE) devices — Routers or switches located at the customer site that connect to the provider's network. CE devices are typically IP routers, but could also be an Asynchronous Transfer Mode (ATM), Frame Relay, or Ethernet switch.

P

Provider (P) routers — Routers within the core of the provider's network that are not connected to any routers at a customer site but are part of the tunnel between pairs of PE routers. P routers support MPLS LSP or LDP functionality, but do not need to support VPN functionality.

Provider edge (PE) routers — Routers in the provider's network that connect to customer edge devices located at customer sites. PE routers support VPN and label functionality. (The label functionality can be provided either by the Resource Reservation Protocol [RSVP] or Label Distribution Protocol [LDP].) Within a single VPN, pairs of PE routers are connected through a tunnel, which can be either an MPLS label-switched path (LSP) or an LDP tunnel.

Types of VPNs

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 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 on page 5
- Layer 3 VPNs on page 5
- VPLS on page 5
- Virtual-Router Routing Instances on page 6

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.

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.

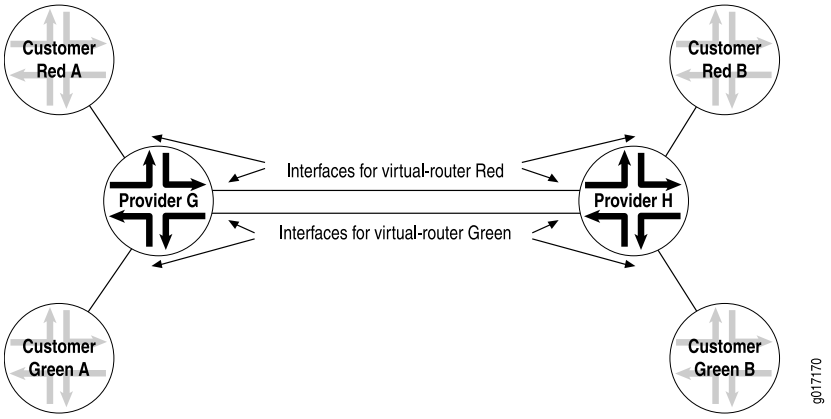
Virtual-Router Routing Instances

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 2 on page 7 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 2: 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 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. For information about how to configure CoS, see the *Junos OS Class of Service Configuration Guide*.

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.

You can configure Layer 2 VPNs, Layer 3 VPNs, VPLS, and Layer 2 circuits within a logical system. For more information about logical systems, see the *Junos OS Routing Protocols Configuration Guide*.

NOTE: Beginning with Junos OS Release 9.3, the logical router feature has been renamed logical system.

All configuration statements, operational commands, show command outputs, error messages, log messages, and SNMP MIB objects that contain the string logical-router or logical-routers have been changed to logical-system and logical-systems, respectively.

VPN Graceful Restart

VPN graceful restart allows a router whose VPN control plane is undergoing a restart to continue to forward traffic while recovering its state from neighboring routers. Without graceful restart, a control plane restart disrupts any VPN services provided by the router.

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.

Redundant Pseudowires for Layer 2 Circuits and VPLS

A redundant pseudowire can act as a backup connection between PE routers and CE devices, maintaining Layer 2 circuit and VPLS services after certain types of failures. This feature can help improve the reliability of certain types of networks (metro for example) where a single point of failure could interrupt service for multiple customers. Redundant pseudowires cannot reduce traffic loss to zero. However, they provide a way to gracefully recover from pseudowire failures in such a way that service can be restarted within a known time limit.

When you configure redundant pseudowires to remote PE routers, you configure one to act as the primary pseudowire over which customer traffic is being transmitted and you configure another pseudowire to act as a backup in the event the primary fails. You configure the two pseudowires statically. A separate label is allocated for the primary and backup neighbors.

For information about how to configure redundant pseudowires, see “Configuring Redundant Pseudowires for Layer 2 Circuits and VPLS” on page 34.

The following sections provide an overview of redundant pseudowires for Layer 2 circuits and VPLS:

Types of Redundant Pseudowire Configurations

You can configure redundant pseudowires for Layer 2 circuits and VPLS in either of the following manners:

- You can configure a single active pseudowire. The PE router configured as the primary neighbor is given preference and this connection is the one used for customer traffic. For the LDP signalling, labels are exchanged for both incoming and outgoing traffic with the primary neighbor. The LDP label advertisement is accepted from the backup neighbor, but no label advertisement is forwarded to it, leaving the pseudowire in an incomplete state. The pseudowire to the backup neighbor is completed only when the primary neighbor fails. The decision to switch between the two pseudowires is made by the device configured with the redundant pseudowires. The primary remote PE router is unaware of the redundant configuration, ensuring that traffic is always switched using just the active pseudowire.
- Alternatively, you can configure two active pseudowires, one to each of the PE routers. Using this approach, control plane signalling is completed and active pseudowires are established with both the primary and backup neighbors. However, the data plane forwarding is done only over a one of the pseudowires (designated as the active pseudowire by the local device). The other pseudowire is on standby. The active pseudowire is preferably established with the primary neighbor and can switch to the backup pseudowire if the primary fails.

  The decision to switch between the active and standby pseudowires is controlled by the local device. The remote PE routers are unaware of the redundant connection, and

so both remote PE routers send traffic to the local device. The local device only accepts traffic from the active pseudowire and drops the traffic from the standby. In addition, the local device only sends traffic to the active pseudowire. If the active pseudowire fails, traffic is immediately switched to the standby pseudowire.

The two configurations available for pseudowire redundancy have the following limitations:

- For the single active pseudowire configuration, it takes more time (compared to the two active pseudowire configuration) to switchover to the backup pseudowire when a failure is detected. This approach requires additional control plane signalling to complete the pseudowire with the backup neighbor and traffic can be lost during the switchover from primary to backup.
- If you configure two active pseudowires, bandwidth is lost on the link carrying the backup pseudowire between the remote PE router and the local device. Traffic is always duplicated over both the active and standby pseudowires. The single active pseudowire configuration does not waste bandwidth in this fashion.
- You cannot enable GRES (graceful Routing Engine switchover) for redundant pseudowires.
- You cannot enable NSR (nonstop active routing) for redundant pseudowires.

Pseudowire Failure Detection

The following events are used to detect a failure (control and data plane) of the pseudowire configured between a local device and a remote PE router and initiates the switch to a redundant pseudowire:

- Manual switchover (user initiated)
- Remote PE router withdraws the label advertisement
- LSP to the remote PE router goes down
- LDP session with the remote PE router goes down
- Local configuration changes
- Periodic pseudowire OAM procedure fails (Layer 2 circuit-based MPLS ping to the PE router fails)

  When you configure a redundant pseudowire between a CE device and a PE router, a periodic (once a minute) ping packet is forwarded through the active pseudowire to verify data plane connectivity. If the ping fails, traffic is automatically switched to the redundant pseudowire.

When a failure is detected, traffic is switched to the redundant pseudowire which is then also designated as the active pseudowire. The switch is nonreversible, meaning that once traffic has been switched to the redundant pseudowire, it remains active unless it also fails unless the switch to the redundant pseudowire is never done unless there is a failure in the currently active pseudowire. For example, a primary pseudowire has failed and traffic has been successfully switched to the redundant pseudowire. After a period of time, the cause of the failure of the primary pseudowire has been resolved and it is now

possible to reestablish the original connection. However, traffic is not switched back to the original pseudowire unless a failure is detected on the now active pseudowire.

VPN Standards

The following IETF RFC and Internet drafts describe VPN features:

- RFC 1918, *Address Allocation for Private Internets*
- RFC 4364, *BGP/MPLS IP Virtual Private Networks (VPNs)*
- RFC 4684, *Constrained Route Distribution for Border Gateway Protocol/MultiProtocol Label Switching (BGP/MPLS) Internet Protocol (IP) Virtual Private Networks (VPNs)*

You can access Internet RFCs and drafts on the IETF website at **http://www.ietf.org**.

CHAPTER 2

Configuring VPNs

Layer 2 virtual private networks (VPNs), Layer 3 VPNs, virtual-router routing instances, and virtual private LAN service (VPLS) use a common infrastructure within Junos and common configuration procedures. This chapter describes the common configuration steps. Complete these configuration steps, regardless of which type of VPN you are configuring, before proceeding to the more specific configuration steps described in other chapters.

This chapter describes the general procedures required to configure Layer 2 VPNs, Layer 3 VPNs, virtual-router routing instances, and VPLS, discussing the following topics:

Configuring the Signaling Protocol on PE Routers in VPNs

For VPNs to function, you must enable a signaling protocol on the provider edge (PE) routers.

NOTE: As with any configuration involving MPLS, you cannot configure any of the core-facing interfaces on the PE routers over dense Fast Ethernet Physical Interface Cards (PICs).

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

- Using LDP for VPN Signaling on page 14
- Using RSVP for VPN Signaling on page 15

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) 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;
  }
}
```

For more information about configuring OSPF and IS-IS, see the *Junos OS Routing Protocols Configuration Guide*.

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 **label-switched-path** and **interface** statements at the **[edit protocols mpls]** hierarchy level:

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

   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 mpls** and **family inet** 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 *Junos OS MPLS Applications Configuration Guide*.

Configuring an IGP on the PE and P Routers

For Layer 2 VPNs, Layer 3 VPNs, virtual-router routing instances, and VPLS to function properly, the service provider's PE and P routers must be able to exchange routing information. To allow them to do this, you must configure either an IGP 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.

For information about configuring IGPs and static routes, see the *Junos OS Routing Protocols Configuration Guide*.

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 at the **[edit protocols bgp group *group-name*]** hierarchy level as follows:

```
[edit protocols]
bgp {
  group group-name {
    type internal;
    local-address ip-address;
    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 (**lo0**) on the local PE router. The IBGP session for the VPN runs through the loopback address. (You must also configure the **lo0** 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 either Layer 2 VPNs and VPLS 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 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 Routing Instances on PE Routers in VPNs

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 on page 19
- Configuring the Description on page 19
- Configuring the Instance Type on page 20
- Configuring Interfaces for VPN Routing on page 20
- Configuring the Route Distinguisher on page 22
- Configuring Automatic Route Distinguishers on page 23

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 on page 21
- Configuring Interfaces for Layer 3 VPNs on page 21

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 the *Junos OS Network Interfaces Configuration Guide*.

Configuring the Route Distinguisher

Each routing instance that you configure on a PE router must have a unique route distinguisher associated with it. VPN and VPLS 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 or VPLS routing instances with the same route distinguisher, the commit fails.

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

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

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***]**

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.

Related Documentation

- Configuring Policies for the VRF Table on PE Routers in VPNs on page 23
- Configuring BGP Route Target Filtering in VPNs on page 29

Configuring Policies for the VRF Table on PE Routers in VPNs

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 on page 24
- Configuring the Route Origin on page 24
- Configuring an Import Policy for the PE Router's VRF Table on page 25
- Configuring an Export Policy for the PE Router's VRF Table on page 27

- Applying Both the VRF Export and the BGP Export Policies on page 28
- Configuring a VRF Target on page 29

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, 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 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 25.
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 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 27.

See "Route Origin for VPNs" on page 53 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 *Junos OS Policy Framework Configuration 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 the *Junos OS Policy Framework Configuration Guide*.

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.

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 *Junos OS Policy Framework Configuration 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 27, 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.

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

- Routes imported into the l3vpn.bgp.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 l3vpn.bgp.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 l3vpn.bgp.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 l3vpn.bgp.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 BGP Route Target Filtering in VPNs

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.

Related Documentation

- BGP Route Target Filtering for VPNs on page 45
- BGP Route Target Filtering for VPNs Overview on page 43
- Route Origin for VPNs on page 53

Configuring Virtual-Router Routing Instances in VPNs

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 how to configure routing protocols, see the *Junos OS Routing Protocols Configuration Guide*.

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 Configuration Guide*.

Configuring Graceful Restart for VPNs

Graceful restart allows a router whose VPN control plane is undergoing a restart to continue to forward traffic while recovering its state from neighboring routers. 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.

To enable VPN graceful restart, include the **graceful-restart** statement:

```
graceful-restart {
    disable;
    restart-duration time-limit;
}
```

To configure graceful restart globally, include the **graceful-restart** statement at the following hierarchy levels:

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

To configure in graceful restart in a particular routing instance, include the **graceful-restart** 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]**

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.

You can include the **restart-duration** option at either the global or routing instance level. The routing instance value overrides the global value if both are configured.

Configuring Redundant Pseudowires for Layer 2 Circuits and VPLS

A redundant pseudowire can act as a backup connection between PE routers and CE devices, maintaining Layer 2 circuit and VPLS services after certain types of failures. This feature can help improve the reliability of certain types of networks (metro for example) where a single point of failure could interrupt service for multiple customers. Redundant pseudowires cannot reduce traffic loss to zero. However, they provide a way to gracefully recover from pseudowire failures in such a way that service can be restarted within a known time limit.

For an overview of how redundant pseudowires work, see "Redundant Pseudowires for Layer 2 Circuits and VPLS" on page 9.

To configure pseudowire redundancy for Layer 2 circuits and VPLS, complete the procedures in the following sections:

Configuring Pseudowire Redundancy on the PE Router

You configure pseudowire redundancy on the PE router acting as the egress for the primary and standby pseudowires using the **backup-neighbor** statement.

To configure pseudowire redundancy on the PE router, include the **backup-neighbor** statement:

```
backup-neighbor {
    community name;
    psn-tunnel-endpoint address;
    standby;
    virtual-circuit-id number;
}
```

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

The **backup-neighbor** statement includes the following configuration options:

- **community**—Specifies the community for the backup neighbor.
- **psn-tunnel-endpoint**—Specifies the endpoint address for the packet switched network (PSN) tunnel on the remote PE router. The PSN tunnel endpoint address is the destination address for the LSP on the remote PE router.
- **standby**—Configures the pseudowire to the specified backup neighbor as the standby. When you configure this statement, traffic flows over both the active and standby pseudowires to the CE device. The CE device drops the traffic from the standby pseudowire, unless the active pseudowire fails. If the active pseudowire fails, the CE device automatically switches to the standby pseudowire.
- **virtual-circuit-id**—Uniquely identifies the primary and standby Layer 2 circuits. This option is configurable for Layer 2 circuits only.

Configuring the Switchover Delay for the Pseudowires

To configure the time the router waits before switching traffic from the failed primary pseudowire to a backup pseudowire, include the **switchover-delay** statement:

switchover-delay *milliseconds*;

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

Configuring a Revert Time for the Redundant Pseudowire

You can specify a revert time for redundant Layer 2 circuit and VPLS pseudowires. When you have configured redundant pseudowires for Layer 2 circuits or VPLS, traffic is switched to the backup pseudowire in the event that the primary pseudowire fails. If you configure a revert time, when the configured time expires traffic is reverted back to the primary pseudowire, assuming the primary pseudowire has been restored.

To configure a revert time for redundant pseudowires, specify the time in seconds using the **revert-time** statement:

revert-time *seconds*;

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

Configuring BFD for VCCV for Layer 2 VPNs, Layer 2 Circuits, and VPLS

Bidirectional Forwarding Detection (BFD) support for virtual circuit connection verification (VCCV) allows you to configure a control channel for a pseudowire, in addition to the corresponding operations and management functions to be used over that control channel. BFD provides a low resource mechanism for the continuous monitoring of the pseudowire data path and for detecting data plane failures.

This feature provides support for asynchronous mode BFD for VCCV as described in draft-ietf-pseudowiree3-vccv-bfd-02.txt, *Bidirectional Forwarding Detection (BFD) for the Pseudowire Virtual Circuit Connectivity Verification (VCCV)*. You can also use a ping operation to detect pseudowire failures. However, the processing resources required for

a ping operation are greater than what is needed for BFD. In addition, BFD is capable of detecting data plane failure faster than VCCV ping. BFD for pseudowires is supported for Layer 2 circuits (LDP-based), Layer 2 VPNs (BGP-based), and VPLS (LDP-based or BGP-based).

To configure OAM and BFD for Layer 2 VPNs, include the **oam** statement and sub-statements at the **[edit routing-instances *routing-instance-name* protocols l2vpn]** hierarchy level:

```
oam {
  ping-interval;
  bfd-liveness-detection {
    detection-time {
      threshold milliseconds;
    }
    minimum-interval milliseconds;
    minimum-receive-interval milliseconds;
    multiplier number;
    no-adaptation;
    transmit-interval {
      minimum-interval milliseconds;
      threshold milliseconds;
    }
    version bfd-protocol-version;
  }
  control-channel {
    pwe3-control-word;
    pseudowire-label-ttl-1;
    router-alert-label;
  }
}
```

For more information about how to configure BFD, see the *Junos OS Routing Protocols Configuration Guide*.

You can configure many of the same OAM statements for VPLS and Layer 2 circuits:

- To enable OAM for VPLS, configure the **oam** statement and substatements at the **[edit routing-instances *routing-instance-name* protocols vpls]** hierarchy level and at the **[edit routing-instances *routing-instance-name* protocols vpls neighbor *address*]** hierarchy level. The **pwe3-control-word** statement configured at the **[edit routing-instances *routing-instance-name* protocols l2vpn oam control-channel]** hierarchy level is not applicable to VPLS configurations.
- To enable OAM for Layer 2 circuits, configure the **oam** statement and substatements at the **[edit protocols l2circuit neighbor *address* interface *interface-name*]** hierarchy level. The **control-channel** statement and sub-statements configured at the **[edit routing-instances routing-instance-name protocols l2vpn oam]** hierarchy level do not apply to Layer 2 circuit configurations.

You can use the **show ldp database extensive** command to display information about the VCCV control channel and the **show bfd session extensive** command to display information about BFD for Layer 2 VPNs, Layer 2 circuits, and VPLS.

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 the *Junos OS Policy Framework Configuration Guide*.

Rewriting 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 *Junos OS Class of Service Configuration Guide*.

Transmitting Nonstandard BPDUs

Circuit cross-connect (CCC) protocol, Layer 2 circuit, and Layer 2 VPN configurations can transmit nonstandard bridge protocol data units (BPDUs) generated by other vendors' equipment. This is the default behavior on all supported PICs and requires no additional configuration.

The following PICs are supported on T Series Core Routers and the M320 Multiservice Edge router and can transmit nonstandard BPDUs:

- 1-port Gigabit Ethernet PIC
- 2-port Gigabit Ethernet PIC
- 4-port Gigabit Ethernet PIC
- 10-port Gigabit Ethernet PIC

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 40).

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*.

Pinging a Layer 2 VPN

To ping a Layer 2 VPN, use one of the following commands:

- **ping mpls l2vpn interface** ***interface-name***

  You ping an interface configured for the Layer 2 VPN on the egress PE router.

- **ping mpls l2vpn instance** ***l2vpn-instance-name*** **local-site-id** ***local-site-id-number*** **remote-site-id** ***remote-site-id-number***

  You ping a combination of the Layer 2 VPN routing instance name, the local site identifier, and the remote site identifier to test the integrity of the Layer 2 VPN connection (specified by the identifiers) between the ingress and egress PE routers.

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.

Pinging a Layer 2 Circuit

To ping a Layer 2 circuit, use one of the following commands:

- **ping mpls l2circuit interface** *interface-name*

  You ping an interface configured for the Layer 2 circuit on the egress PE router.

- **ping mpls l2circuit virtual-circuit neighbor** <*prefix*> <*virtual-circuit-id*>

  You ping a combination of the IPv4 prefix and the virtual circuit identifier on the egress PE router to test the integrity of the Layer 2 circuit between the ingress and egress PE routers.

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 1.1.1.1 prefix 2.2.2.2 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 difference 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>
```

You ping a combination of the routing instance name, the destination MAC address, and the source IP address. When you enter 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.

Configuring Path MTU Checks for VPNs

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

and T Series routers. 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 System Basics Configuration Guide*.

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

- Enabling Path MTU Checks for a VPN Routing Instance on page 41
- Assigning an IP Address to the VPN Routing Instance on page 41

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 193 for details.

Enabling Unicast Reverse-Path Forwarding Check for VPNs

IP spoofing may occur during a denial-of-service (DoS) attack. IP spoofing allows an intruder to pass IP packets to a destination as genuine traffic, when in fact the packets are not actually meant for the destination. This type of spoofing is harmful because it consumes the destination’s resources.

Unicast reverse-path forwarding (RPF) check is a tool to reduce forwarding of IP packets that may be spoofing an address. A unicast RPF check performs a route table lookup on

an IP packet's source address, and checks the incoming interface. The router determines whether the packet is arriving from a path that the sender would use to reach the destination. If the packet is from a valid path, the router forwards the packet to the destination address. If it is not from a valid path, the router discards the packet. Unicast RPF is supported for the IPv4 and IPv6 protocol families, as well as for the virtual private network (VPN) address family. You can also enable unicast RPF within a VPN routing instance.

To enable unicast RPF check, include the **unicast-reverse-path** statement:

```
unicast-reverse-path (active-paths | feasible-paths);
```

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

To consider only active paths during the unicast RPF check, include the **active-paths** option. To consider all feasible paths during the unicast RPF check, include the **feasible-paths** option.

The **unicast-reverse-path** statement is documented in greater detail in the *Junos OS Routing Protocols Configuration Guide* and the *Junos OS Network Interfaces Configuration Guide*.

CHAPTER 3

VPN Examples

The following examples illustrate how to configure BGP route target filtering for virtual private networks (VPNs):

BGP Route Target Filtering for VPNs Overview

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:

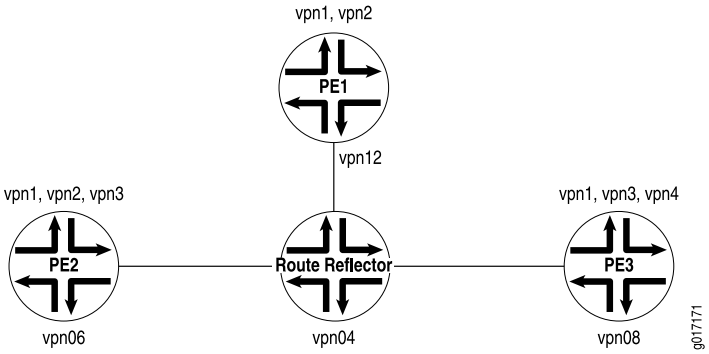
- **200:200:103/96**—Tells the local router that routes tagged with this community (**200:103**) should be advertised to peer **10.255.14.174** through **t3-0/0/0.0**
- **200:200:104/96**—Tells the local router that routes tagged with this community (**200:104**) should be advertised to peer **10.255.14.174**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

BGP Route Target Filtering for VPNs

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 3 on page 45 illustrates the topology for a network configured with BGP route target filtering for a group of VPNs.

Figure 3: 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 on page 46
- Configure BGP Route Target Filtering on Router PE2 on page 47
- Configure BGP Route Target Filtering on the Route Reflector on page 50
- Configure BGP Route Target Filtering on Router PE3 on page 51

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 200;
}
```

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 200;
         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:200:101;
      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:200:102;
    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

200:200:101/96
                   *[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
200:200:102/96
                   *[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
200:200:103/96
                   *[BGP/170] 00:27:30, localpref 100, from
10.255.14.174
                      AS path: I
                    > via t3-0/0/0.0
200:200:104/96
                   *[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 200;
}
```

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 200;
         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:200:101;
      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:200:102;
      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

 200:200:101/96
                    *[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
 200:200:102/96
                    *[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
 200:200:103/96
                    *[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
 200:200:104/96
                    *[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 200;
}
```

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 200;
         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

 200:200:101/96
                    *[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
200:200:102/96
                   *[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
200:200:103/96
                   *[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
200:200:104/96
                   *[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 200;
}
```

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 200;
      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:200:101;
    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:200:103;
    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:200:104;
    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:

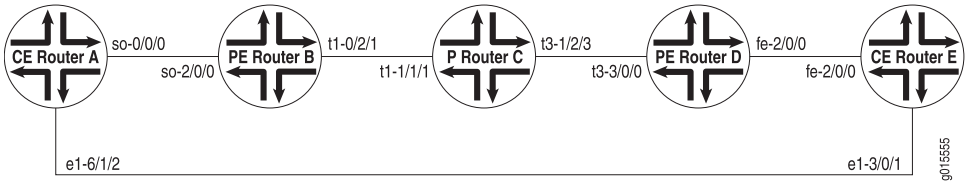
```
user@host> show route table bgp.rtarget.0
bgp.rtarget.0: 4 destinations, 7 routes (4 active, 0 holddown, 0 hidden)
 + = Active Route, - = Last Active, * = Both

 200:200:101/96
                    *[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
 200:200:102/96
                    *[BGP/170] 00:29:29, localpref 100, from
 10.255.14.174
                       AS path: I
                     > via t3-0/0/1.0
 200:200:103/96
                    *[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
 200:200:104/96
                    *[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
```

Route Origin for VPNs

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 4 on page 54.

Figure 4: 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 on page 54
- Configuring the Community on CE Router A on page 54
- Applying the Policy Statement on CE Router A on page 55
- Configuring the Policy on PE Router D on page 55
- Configuring the Community on PE Router D on page 56
- Applying the Policy on PE Router D on page 56

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;
      }
    }
  }
}
```

CHAPTER 4

Summary of VPN Configuration Statements

This chapter summarizes the statements used in the configuration of virtual private networks (VPNs) and virtual private LAN service (VPLS). The statements are organized alphabetically.

Statements configured at the **[edit routing-instances]** and the **[edit protocols]** hierarchy levels are explained in complete detail in the *Junos OS Routing Protocols Configuration Guide*.

Statements configured at the **[edit policy-options]** hierarchy level are explained in complete detail in the *Junos OS Policy Framework Configuration Guide*.

Statements configured at the **[edit interfaces]** hierarchy level are explained in complete detail in the *Junos OS Network Interfaces Configuration Guide*.

aggregate-label

Syntax

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

Hierarchy Level

[edit logical-systems *logical-system-name* protocols bgp family inet labeled-unicast],
[edit logical-systems *logical-system-name* protocols bgp family inet6 labeled-unicast],
[edit logical-systems *logical-system-name* protocols bgp family inet-vpn unicast],
[edit logical-systems *logical-system-name* protocols bgp family inet-vpn6 unicast],
[edit protocols bgp family inet labeled-unicast],
[edit protocols bgp family inet6 labeled-unicast],
[edit protocols bgp family inet-vpn unicast],
[edit protocols bgp family inet6-vpn unicast]

Release Information

Statement introduced before Junos OS Release 7.4.

Description

Specify matching criteria (in the form of a community) such that all routes which match are assigned the same VPN label, selected from one of the several routes in the set defined by this criteria. This reduces the number of VPN labels that the router must consider, and aggregates the received labels.

Options

community *community-name*—Specify the name of the community to which to apply the aggregate label.

Required Privilege Level

routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation

- Configuring Aggregate Labels for VPNs on page 37

backup-neighbor

Syntax

```
backup-neighbor address {
    community name;
    psn-tunnel-endpoint address;
    standby;
    virtual-circuit-id number;
}
```

Hierarchy Level

[edit logical-systems *logical-system-name* protocols l2circuit neighbor *address* interface *interface-name*],
[edit logical-systems *logical-system-name* routing-instances *routing-instance-name* protocols vpls neighbor *address*],
[edit protocols l2circuit neighbor *address* interface *interface-name*],
[edit routing-instances *routing-instance-name* protocols vpls neighbor *address*]

Release Information

Statement introduced in Junos OS Release 9.2.

Description

Configures pseudowire redundancy for Layer 2 circuits and VPLS. A redundant pseudowire can act as a backup connection between a PE router and a CE device, maintaining Layer 2 circuit and VPLS services after certain types of failures. This feature can help improve the reliability of certain types of networks where a single point of failure could interrupt service for multiple customers.

Options

community—Specifies the community for the backup neighbor.

psn-tunnel-endpoint—Specifies the endpoint address for the packet switched network (PSN) tunnel on the remote PE router. The PSN tunnel endpoint address is the destination address for the LSP on the remote PE router.

standby—Configures the pseudowire to the specified backup neighbor as the standby. When you configure this statement, traffic flows over both the active and standby pseudowires to the CE device. The CE device drops the traffic from the standby pseudowire, unless the active pseudowire fails. If the active pseudowire fails, the CE device automatically switches to the standby pseudowire.

virtual-circuit-id—Uniquely identifies the primary and standby Layer 2 circuits. This option is configurable for Layer 2 circuits only.

Required Privilege Level

routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation

- **community on page 676**
- **psn-tunnel-endpoint on page 692**
- **virtual-circuit-id on page 695**
- Configuring Pseudowire Redundancy on the PE Router on page 34

description

Syntax description *text*;

Hierarchy Level [edit logical-systems *logical-system-name* routing-instances *routing-instance-name*], [edit routing-instances *routing-instance-name*]

Release Information Statement introduced before Junos OS Release 7.4.

Description Describe the VPN or VPLS routing instance.

Options *text*—Provide a text description. If the text includes one or more spaces, enclose the text 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 operation.

Required Privilege Level routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation

- Configuring the Description on page 19

family route-target

Syntax

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

Hierarchy Level

[edit logical-systems *logical-system-name* protocols bgp group *group-name*],
[edit logical-systems *logical-system-name* protocols bgp group *group-name* neighbor *address*],
[edit protocols bgp group *group-name*],
[edit protocols bgp group *group-name* neighbor *address*]

Release Information

Statement introduced before Junos OS Release 7.4.

Description

Enable BGP route target filtering on the Layer 3 VPN.

Options

advertise-default—Cause 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 provider edge (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.

external-paths *number*—Cause the router to advertise the VPN routes that reference a given route target. The number you specify with the **external-paths** statement determines the number of external peer routers (currently advertising that route target) that receive the VPN routes. The default value is **1**.

prefix-limit *number*—The number of prefixes that can be received from a peer router.

Required Privilege Level

routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation

- Configuring BGP Route Target Filtering for VPNs on page 30

graceful-restart

Syntax

```
graceful-restart {
  disable;
  restart-duration time-limit;
}
```

Hierarchy Level

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

Release Information

Statement introduced before Junos OS Release 7.4.

Description

Allow a router whose VPN control plane is undergoing a restart to continue to forward traffic while recovering its state from neighboring routers.

Options

disable—Disable graceful restart.

restart-duration *time-limit*—Grace period for graceful restart, in seconds.

Default: 300 seconds

Range: 1 through 600 seconds

Required Privilege Level

routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation

- Configuring Graceful Restart for VPNs on page 33

instance-type

Syntax instance-type *type*;

Hierarchy Level [edit logical-systems *logical-system-name* routing-instances *routing-instance-name*], [edit routing-instances *routing-instance-name*]

Release Information Statement introduced before Junos OS Release 7.4.
Statement introduced in Junos OS Release 9.2 for EX Series switches.

Description Define the type of routing instance.

Options *type*—Can be one of the following:

- **l2vpn**—Enable a Layer 2 VPN on the routing instance. You must configure the **interface**, **route-distinguisher**, **vrf-import**, and **vrf-export** statements for this type of routing instance.
- **virtual-router**—Enable a virtual router routing instance. You must configure the **interface** statement for this type of routing instance. You do not need to configure the **route-distinguisher**, **vrf-import**, and **vrf-export** statements.
- **vpls**—Enable VPLS on the routing instance. You must configure the **interface**, **route-distinguisher**, **vrf-import**, and **vrf-export** statements for this type of routing instance.
- **vrf**—VPN routing and forwarding (VRF) instance. Required to create a Layer 3 VPN. Create a VRF table (***instance-name*.inet.0**) that contains the routes originating from and destined for a particular Layer 3 VPN. You must configure the **interface**, **route-distinguisher**, **vrf-import**, and **vrf-export** statements for this type of routing instance.

Required Privilege Level routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation

- Example: Using Virtual Routing Instances to Route Among VLANs on EX Series Switches
- Configuring the Instance Type on page 20
- Configuring Virtual Routing Instances (CLI Procedure)

interface

Syntax interface *interface-name*;

Hierarchy Level [edit logical-systems *logical-system-name* routing-instances *routing-instance-name*],
[edit routing-instances *routing-instance-name*]

Release Information Statement introduced before Junos OS Release 7.4.
Statement introduced in Junos OS Release 9.2 for EX Series switches.

Description Interface over which the VPN traffic travels between the PE router or switch and customer edge (CE) router or switch. You configure the interface on the PE router or switch. If the value **vrf** is specified for the **instance-type** statement included in the routing instance configuration, this statement is required.

Options *interface-name*—Name of the interface.

Required Privilege Level routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation
- **instance-type on page 63**
- Configuring Interfaces for VPN Routing on page 20

no-forwarding

Syntax no-forwarding;

Hierarchy Level [edit logical-systems *logical-system-name* protocols ldp],
[edit logical-systems *logical-system-name* routing-instances *routing-instance-name* protocols ldp],
[edit protocols ldp],
[edit routing-instances *routing-instance-name* protocols ldp]

Release Information Statement introduced before Junos OS Release 7.4.

Description Do not add ingress routes to the **inet.0** routing table even if **traffic-engineering bgp-igp** (configured at the **[edit protocols mpls]** hierarchy level) is enabled.

Default The **no-forwarding** statement is disabled. Ingress routes are added to the **inet.0** routing table instead of the **inet.3** routing table when **traffic-engineering bgp-igp** is enabled.

Required Privilege Level routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation
- Configuring a Routing Protocol Between the Service Provider Routers on page 32

revert-time

Syntax revert-time *seconds*;

Hierarchy Level [edit logical-systems *logical-system-name* protocols l2circuit neighbor *address* interface *interface-name*],
[edit logical-systems *logical-system-name* routing-instances *routing-instance-name* protocols vpls neighbor *address*],
[edit protocols l2circuit neighbor *address* interface *interface-name*],
[edit routing-instances *routing-instance-name* protocols vpls neighbor *address*]

Release Information Statement introduced in Junos OS Release 10.2.

Description Specifies a revert time for redundant Layer 2 circuits and VPLS pseudowires. When you have configured redundant pseudowires for Layer 2 circuits or VPLS, traffic is switched to the backup connection in the event that the primary connection fails. If you configure a revert time, when the configured time expires traffic is reverted to the primary path, assuming the primary path has been restored.

Options *seconds*—Revert time in seconds.

Range: 0 through 65,535 seconds

Default: 5 seconds

Required Privilege Level routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation

- Configuring Redundant Pseudowires for Layer 2 Circuits and VPLS on page 34

route-distinguisher

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

Hierarchy Level [edit logical-systems *logical-system-name* routing-instances *routing-instance-name*], [edit routing-instances *routing-instance-name*]

Release Information Statement introduced before Junos OS Release 7.4.
Statement introduced in Junos OS Release 11.1 for EX Series switches.

Description Specify an identifier attached to a route, enabling you to distinguish to which VPN the route belongs. Each routing instance must have a unique route distinguisher associated with it. The route distinguisher is used to place bounds around a VPN so that the same IP address prefixes can be used in different VPNs without having them overlap. If the instance type is **vrf**, the **route-distinguisher** statement is required.

Options ***as-number:number*—*as-number*** is an assigned AS number and ***number*** is any 2-byte for 4-byte value. The AS number can be from 1 through 4,294,967,295. If the AS number is a 2-byte value, the administrative number is a 4-byte value. If the AS number is 4-byte value, the administrative number is a 2-byte value. A route distinguisher consisting of a 4-byte AS number and a 2-byte administrative number is defined as a type 2 route distinguisher in RFC 4364 *BGP/MPLS IP Virtual Private Networks*.

NOTE: In Junos OS Release 9.1 and later, the numeric range for AS numbers is extended to provide BGP support for 4-byte AS numbers, as defined in RFC 4893, *BGP Support for Four-octet AS Number Space*. All releases of the Junos OS support 2-byte AS numbers. To configure a route distinguisher that includes a 4-byte AS number, append the letter "L" to the end of the number. For example, a route distinguisher with the 4-byte AS number 7,765,000 and an administrative number of 1,000 is represented as 77765000L:1000.

In Junos OS Release 9.2 and later, you can also configure a 4-byte AS number using the AS-dot notation format of two integer values joined by a period: *<16-bit high-order value in decimal>.<16-bit low-order value in decimal>*. For example, the 4-byte AS number of 65,546 in plain-number format is represented as 1.10 in the AS-dot notation format.

ip-address:number*—*ip-address is an IP address in your assigned prefix range (a 4-byte value) and ***number*** is any 2-byte value.

Required Privilege Level routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation

- Configuring the Route Distinguisher on page 22
- Configuring Route Distinguishers for Routing Instances
- Configuring an MPLS-Based Layer 2 VPN (CLI Procedure)

- Configuring an MPLS-Based Layer 3 VPN (CLI Procedure)
- Understanding 4-Byte AS Numbers and Route Distinguishers in the *Using 4-Byte Autonomous System Numbers in BGP Networks Technology Overview*

route-distinguisher-id

Syntax route-distinguisher-id *ip-address*;

Hierarchy Level [edit logical-systems *logical-system-name* routing-options],
[edit routing-options]

Release Information Statement introduced before Junos OS Release 7.4.

Description Automatically assign a route distinguisher to the routing instance. If you 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**.

Options ***ip-address***—Address for routing instance.

Required Privilege Level routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation
- Configuring the Route Distinguisher on page 22

switchover-delay

Syntax switchover-delay *milliseconds*;

Hierarchy Level [edit logical-systems *logical-system-name* protocols l2circuit neighbor *address* interface *interface-name*],
[edit logical-systems *logical-system-name* routing-instances *routing-instance-name* protocols vpls neighbor *address*],
[edit protocols l2circuit neighbor *address* interface *interface-name*],
[edit routing-instances *routing-instance-name* protocols vpls neighbor *address*],

Release Information Statement introduced in Junos OS Release 9.2.

Description After the primary pseudowire goes down, specifies the delay (in milliseconds) to wait before the backup pseudowire takes over. You configure this statement for each backup neighbor configuration to adjust the switchover time after a failure is detected.

Options *milliseconds*—Specify the time to wait before switching to the backup pseudowire after the primary pseudowire fails.

Default: 10,000 milliseconds

Range: 0 through 180,000 milliseconds

Required Privilege Level routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation

- Configuring the Switchover Delay for the Pseudowires on page 35

unicast-reverse-path

Syntax unicast-reverse-path (active-paths | feasible-paths);

Hierarchy Level [edit logical-systems *logical-system-name* routing-instances *routing-instance-name* routing-options forwarding-table],
[edit logical-systems *logical-system-name* routing-options forwarding-table],
[edit routing-instances *routing-instance-name* routing-options forwarding-table],
[edit routing-options forwarding-table]

Release Information Statement introduced before Junos 7.4. Statement added at the **[edit routing-instances]** hierarchy level in Junos 8.3.

Description Enable unicast reverse-path-forwarding check.

Options **active-paths**—Consider only active paths during the unicast RPF check.

feasible-paths—Consider all feasible paths during the unicast RPF check.

Required Privilege Level routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation
- Enabling Unicast Reverse-Path Forwarding Check for VPNs on page 41

vpn-apply-export

Syntax vpn-apply-export;

Hierarchy Level [edit logical-systems *logical-system-name* protocols bgp],
[edit logical-systems *logical-system-name* protocols bgp group *group-name*],
[edit logical-systems *logical-system-name* protocols bgp group *group-name* neighbor *neighbor*],
[edit protocols bgp],
[edit protocols bgp group *group-name*],
[edit protocols bgp group *group-name* neighbor *neighbor*]

Release Information Statement introduced before Junos OS Release 7.4.

Description Apply both the VRF export and BGP group or neighbor export policies (VRF first, then BGP) before routes from the **vrf** or **l2vpn** routing tables are advertised to other PE routers.

Required Privilege Level routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation
- Applying Both the VRF Export and the BGP Export Policies on page 28

vrf-export

Syntax vrf-export [*policy-names*];

Hierarchy Level [edit logical-systems *logical-system-name* routing-instances *routing-instance-name*], [edit routing-instances *routing-instance-name*]

Release Information Statement introduced before Junos OS Release 7.4.
Statement introduced in Junos OS Release 11.1 for EX Series switches.

Description Specify how routes are exported from the local PE router's VRF table (***routing-instance-name*.inet.0**) to the remote PE router. If the value **vrf** is specified for the **instance-type** statement included in the routing instance configuration, this statement is required.

You can configure multiple export policies on the PE router or PE switch (EX8200 switch only).

Options ***policy-names***—Names for the export policies.

Required Privilege Level routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation

- **instance-type on page 63**
- Configuring an Export Policy for the PE Router's VRF Table on page 27

vrf-import

Syntax vrf-import [*policy-names*];

Hierarchy Level [edit logical-systems *logical-system-name* routing-instances *routing-instance-name*], [edit routing-instances *routing-instance-name*]

Release Information Statement introduced before Junos OS Release 7.4.
Statement introduced in Junos OS Release 11.1 for EX Series switches.

Description Specify how routes are imported into the VRF table (***routing-instance-name*.inet.0**) of the local provider edge (PE) router or switch (EX8200 only) from the remote PE. If the value **vrf** is specified for the **instance-type** statement included in the routing instance configuration, this statement is required.

You can configure multiple import policies on the PE router or PE switch (EX8200 switch only).

Options ***policy-names***—Names for the import policies.

Required Privilege Level routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation
- **instance-type on page 63**
- Configuring an Import Policy for the PE Router's VRF Table on page 25

vrf-mtu-check

Syntax vrf-mtu-check;

Hierarchy Level [edit chassis]

Release Information Statement introduced before Junos OS Release 7.4.

Description Enable path checks on the outgoing interface for unicast traffic routed on a VRF or virtual-router routing instance.

Required Privilege Level interface—To view this statement in the configuration.
interface-control—To add this statement to the configuration.

Related Documentation
- Configuring Path MTU Checks for VPNs on page 40

vrf-target

Syntax

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

Hierarchy Level [edit logical-systems *logical-system-name* routing-instances *routing-instance-name*], [edit routing-instances *routing-instance-name*]

Release Information Statement introduced before Junos OS Release 7.4.
Statement introduced in Junos OS Release 11.1 for EX Series switches.

Description Specify a VRF target community. If you configure the ***community*** option only, default VRF import and export policies are generated that accept and tag routes with the specified target community. 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.

You can still create more complex policies by explicitly configuring VRF import and export policies using the **import** and **export** options.

Options ***community***—Community name.

import *community-name*—Communities accepted from neighbors.

export *community-name*—Communities sent to neighbors.

Required Privilege Level routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation

- Configuring a VRF Target on page 29

PART 2

Layer 2 VPNs

CHAPTER 5

Layer 2 VPN Overview

This chapter discusses the following topics that provide background information about Layer 2 VPNs:

Layer 2 VPN Overview

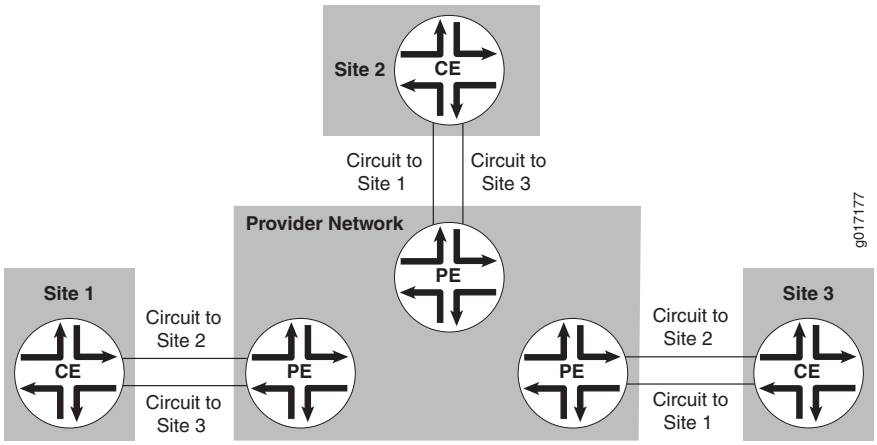
Implementing a Layer 2 VPN on a router is similar to implementing a VPN using a Layer 2 technology such as Asynchronous Transfer Mode (ATM) or Frame Relay. However, for a Layer 2 VPN on a router, traffic is forwarded to the router in a Layer 2 format. It is carried by MPLS over the service provider's 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 customer edge (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 provider edge (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 will need 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.

Customers need to know only which VPN interfaces connect to which of their own sites. Figure 5 on page 76 illustrates a Layer 2 VPN in which each site has a VPN interface linked to each of the other customer sites.

Figure 5: Layer 2 VPN Connecting CE Routers



Implementing a Layer 2 MPLS VPN includes the following benefits:

- Service providers do not have to invest in separate Layer 2 equipment to provide Layer 2 VPN service. A Layer 2 MPLS VPN allows you to provide Layer 2 VPN service over an existing IP and MPLS backbone.
- You can configure the PE 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 might want Layer 2 VPN connections with their service provider instead of a Layer 3 VPN.

Layer 2 VPN Standards

The Junos OS substantially supports the following Layer 2 VPN Internet draft: draft-kompella-ppvpn-l2vpn-03.txt, *Layer 2 VPN Over Tunnels*.

You can access Internet RFCs and drafts on the IETF website at **http://www.ietf.org**.

CHAPTER 6

Configuring Layer 2 VPNs

The following sections describe how to configure Layer 2 VPNs:

Introduction to Configuring Layer 2 VPNs

To configure Layer 2 virtual private network (VPN) functionality, you must enable Layer 2 VPN support on the provider edge (PE) router. You must also configure PE routers to distribute routing information to the other PE routers in the VPN and configure the circuits between the PE routers and the customer edge (CE) routers.

Each Layer 2 VPN is configured under a routing instance of type **l2vpn**. An **l2vpn** routing instance can transparently carry Layer 3 traffic across the service provider's network. As with other routing instances, all logical interfaces belonging to a Layer 2 VPN routing instance are listed under that instance.

The configuration of the CE routers is not relevant to the service provider. The CE routers need to provide only appropriate Layer 2 circuits (with appropriate circuit identifiers, such as data-link connection identifier [DLCI], virtual path identifier/virtual channel identifier [VPI/VCI], or virtual LAN [VLAN] ID) to send traffic to the PE router.

To configure Layer 2 VPNs, include the following statements:

```
description text;
instance-type l2vpn;
interface interface-name;
route-distinguisher (as-number:id| ip-address:id);
vrf-export [ policy-names ];
vrf-import [ policy-names ];
vrf-target {
   community;
   import community-name;
   export community-name;
```

```
}
protocols {
  l2vpn {
    (control-word | no-control-word);
    encapsulation-type type;
    site site-name {
      interface interface-name {
        description text;
        remote-site-id remote-site-id;
      }
      site-identifier identifier;
      site-preference preference-value {
        backup;
        primary;
      }
    }
    traceoptions {
      file filename <files number> <size size> <world-readable | no-world-readable>;
      flag flag <flag-modifier> <disable>;
    }
  }
}
```

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***]**

For Layer 2 VPNs, only some of the statements in the **[edit routing-instances]** hierarchy are valid. For the full hierarchy, see the *Junos OS Routing Protocols Configuration Guide*.

In addition to these statements, you must configure MPLS label-switched paths (LSPs) between the PE routers, IBGP sessions between the PE routers, and an interior gateway protocol (IGP) on the PE and provider (P) routers. You must also configure the statements that are required for all types of VPN configuration.

By default, Layer 2 VPNs are disabled.

Many of the configuration procedures for Layer 2 VPNs are identical to the procedures for Layer 3 VPNs and virtual private LAN service (VPLS).

Configuring the Local Site on PE Routers in Layer 2 VPNs

For each local site, the PE router advertises a set of VPN labels to the other PE routers servicing the Layer 2 VPN. The VPN labels constitute a single block of contiguous labels; however, to allow for reprovisioning, more than one such block can be advertised. Each label block consists of a label base, a range (the size of the block), and a remote site ID that identifies the sequence of remote sites that connect to the local site using this label block (the remote site ID is the first site identifier in the sequence). The encapsulation type is also advertised along with the label block.

The following sections explain how to configure the connections to the local site on the PE router:

Configuring a Layer 2 VPN Routing Instance

To configure a Layer 2 VPN on your network, configure a Layer 2 VPN routing instance on the PE router by including the **l2vpn** statement:

```
l2vpn {
  (control-word | no-control-word);
  encapsulation-type type;
  traceoptions {
    file filename <files number> <size size> <world-readable | no-world-readable>;
    flag flag <flag-modifier> <disable>;
  }
  site site-name {
    site-identifier identifier;
    site-preference preference-value {
      backup;
      primary;
    }
    interface interface-name {
      description text;
      remote-site-id remote-site-id;
    }
  }
}
```

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: You cannot configure a routing protocol (OSPF, RIP, IS-IS or BGP) inside a Layer 2 VPN routing instance (instance-type l2vpn). The Junos CLI disallows this configuration.

Instructions for how to configure the remaining statements are included in the sections that follow.

Configuring the Site

All the Layer 2 circuits provisioned for a local site are listed as the set of logical interfaces (specified by including the **interface** statement) within the **site** statement.

On each PE router, you must configure each site that has a circuit to the PE router. To do this, include the **site** statement:

```
site site-name {
    site-identifier identifier;
    site-preference preference-value {
        backup;
        primary;
    }
    interface interface-name {
        description text;
        remote-site-id remote-site-ID;
    }
}
```

You can include this statement at the following hierarchy levels:

- **[edit routing-instances *routing-instance-name* protocols l2vpn]**
- **[edit logical-systems *logical-system-name* routing-instances *routing-instance-name* protocols l2vpn]**

You must configure the following for each site:

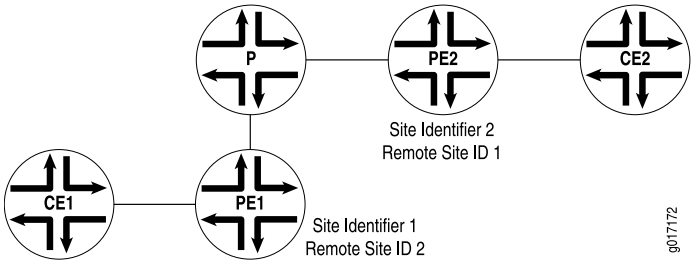
- ***site-name***—Name of the site.
- **site-identifier *identifier***—Unsigned 16-bit number greater than zero that uniquely identifies the local Layer 2 VPN site. The site identifier corresponds to the remote site ID configured on another site within the same VPN.
- **interface *interface-name***—The name of the interface and, optionally, a remote site ID for remote site connections. See “Configuring the Remote Site ID” on page 80.

Configuring the Remote Site ID

The remote site ID allows you to configure a sparse Layer 2 VPN topology. A sparse topology means that each site does not have to connect to all the other sites in the VPN; thus it is unnecessary to allocate circuits for all the remote sites. Remote site IDs are particularly important if you configure a topology more complicated than full-mesh, such as a hub-and-spoke topology.

The remote site ID (configured with the **remote-site-id** statement) corresponds to the site ID (configured with the **site-identifier** statement) configured at a separate site. Figure 6 on page 81 illustrates the relationship between the site identifier and the remote site ID.

Figure 6: Relationship Between the Site Identifier and the Remote Site ID



As illustrated by the figure, the configuration for Router PE1 connected to Router CE1 is as follows:

```
site-identifier 1;
interface so-0/0/0 {
  remote-site-id 2;
}
```

The configuration for Router PE2 connected to Router CE2 is as follows:

```
site-identifier 2;
interface so-0/0/1 {
  remote-site-id 1;
}
```

The remote site ID (**2**) on Router PE1 corresponds to the site identifier (**2**) on Router PE2. On Router PE2, the remote site ID (**1**) corresponds to the site identifier (**1**) on Router PE1.

To configure the remote site ID, include the **remote-site-id** statement:

```
remote-site-id remote-site-id;
```

You can include this statement at the following hierarchy levels:

- **[edit routing-instances** *routing-instance-name* **protocols l2vpn site** *site-name* **interface** *interface-name***]**
- **[edit logical-systems** *logical-system-name* **routing-instances** *routing-instance-name* **protocols l2vpn site** *site-name* **interface** *interface-name***]**

If you do not explicitly include the **remote-site-id** statement for the interface configured at the **[edit routing-instances** *routing-instance-name* **protocols l2vpn site** *site-name***]** hierarchy level, a remote site ID is assigned to that interface.

The remote site ID for an interface is automatically set to 1 higher than the remote site ID for the previous interface. The order of the interfaces is based on their **site-identifier** statements. For example, if the first interface in the list does not have a remote site ID, its ID is set to 1. The second interface in the list has its remote site ID set to 2, and the third has its remote site ID set to 3. The remote site IDs of any interfaces that follow are incremented in the same manner if you do not explicitly configure them.

Configuring the Encapsulation Type

The encapsulation type you configure at each Layer 2 VPN site varies depending on which Layer 2 protocol you choose to configure. If you configure **ethernet-vlan** as the encapsulation type, you need to use the same protocol at each Layer 2 VPN site.

You do not need to use the same protocol at each Layer 2 VPN site if you configure any of the following encapsulation types:

- **atm-aal5**—Asynchronous Transfer Mode (ATM) Adaptation Layer (AAL5)
- **atm-cell**—ATM cell relay
- **atm-cell-port-mode**—ATM cell relay port promiscuous mode
- **atm-cell-vc-mode**—ATM virtual circuit (VC) cell relay nonpromiscuous mode
- **atm-cell-vp-mode**—ATM virtual path (VP) cell relay promiscuous mode
- **cisco-hdlc**—Cisco Systems-compatible High-Level Data Link Control (HDLC)
- **ethernet**—Ethernet
- **ethernet-vlan**—Ethernet virtual LAN (VLAN)
- **frame-relay**—Frame Relay
- **frame-relay-port-mode**—Frame Relay port mode
- **interworking**—Layer 2.5 interworking VPN
- **ppp**—Point-to-Point Protocol (PPP)

If you configure different protocols at your Layer 2 VPN sites, you need to configure a translational cross-connect (TCC) encapsulation type. For more information, see “Configuring TCC Encapsulation for Layer 2 VPNs and Layer 2 Circuits” on page 85.

To configure the Layer 2 protocol accepted by the PE router, specify the encapsulation type by including the **encapsulation-type** statement:

encapsulation-type *type*;

You can include this statement at the following hierarchy levels:

- **[edit routing-instances** *routing-instance-name* **protocols l2vpn]**
- **[edit logical-systems** *logical-system-name* **routing-instances** *routing-instance-name* **protocols l2vpn]**

Configuring a Site Preference and Layer 2 VPN Multihoming

You can specify the preference value advertised for a particular Layer 2 VPN site. The site preference value is encoded in the BGP local preference attribute. When a PE router receives multiple advertisements with the same CE device identifier, the advertisement with the highest local preference value is preferred.

You can also use the **site-preference** statement to enable multihoming for Layer 2 VPNs. Multihoming allows 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.

To configure a site preference for a Layer 2 VPN, include the **site-preference** statement:

```
site-preference preference-value {
    backup;
    primary;
}
```

You can include this statement at the following hierarchy levels:

- **[edit routing-instances *routing-instance-name* protocols l2vpn site *site-name*]**
- **[edit logical-systems *logical-system-name* routing-instances *routing-instance-name* protocols l2vpn site *site-name*]**

You can also specify either the **backup** option or the **primary** option for the **site-preference** statement. The backup option specifies the preference value as 1, the lowest possible value, ensuring that the Layer 2 VPN site is the least likely to be selected. The primary option specifies the preference value as 65,535, the highest possible value, ensuring that the Layer 2 VPN site is the most likely to be selected.

For Layer 2 VPN multihoming configurations, specifying the **primary** option for a Layer 2 VPN site designates the connection from the PE router to the CE device as the preferred connection if the CE device is also connected to another PE router. Specifying the **backup** option for a Layer 2 VPN site designates the connection from the PE router to the CE device as the secondary connection if the CE device is also connected to another PE router.

Tracing Layer 2 VPN Traffic and Operations

To trace Layer 2 VPN protocol traffic, specify options for the **traceoptions** statement in the Layer 2 VPN configuration:

```
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 l2vpn]**
- **[edit logical-systems *logical-system-name* routing-instances *routing-instance-name* protocols l2vpn]**

The following trace flags display the operations associated with Layer 2 VPNs:

- **all**—All Layer 2 VPN tracing options.
- **connections**—Layer 2 connections (events and state changes).
- **error**—Error conditions.

- **general**—General events.
- **nlri**—Layer 2 advertisements received or sent by means of the BGP.
- **normal**—Normal events.
- **policy**—Policy processing.
- **route**—Routing information.
- **state**—State transitions.
- **task**—Routing protocol task processing.
- **timer**—Routing protocol timer processing.
- **topology**—Layer 2 VPN topology changes caused by reconfiguration or advertisements received from other PE routers using BGP.

Disabling Normal TTL Decrementing for VPNs

To diagnose networking problems related to VPNs, it can be useful to disable normal time-to-live (TTL) decrementing. In Junos, you can do this with the **no-propagate-ttl** and **no-decrement-ttl** statements. However, when you are tracing VPN traffic, only the **no-propagate-ttl** statement is effective.

For the **no-propagate-ttl** statement to have an effect on VPN behavior, you need to clear the PE-router-to-PE-router BGP session, or disable and then enable the VPN routing instance.

For more information about the **no-propagate-ttl** and **no-decrement-ttl** statements, see the *Junos OS MPLS Applications Configuration Guide*.

Configuring CCC Encapsulation for Layer 2 VPNs

You need to specify a circuit cross-connect (CCC) encapsulation type for each PE-router-to-CE-router interface running a Layer 2 VPN. This encapsulation type should match the encapsulation type configured under the routing instance. For information about how to configure the encapsulation type under the routing instance, see “Configuring the Encapsulation Type” on page 82.

NOTE: A Layer 2 VPN or Layer 2 circuit is not supported if the PE-router-to-P-router interface has VLAN-tagging enabled and uses a nonenhanced Flexible PIC Concentrator (FPC).

For Layer 2 VPNs, you need to configure the CCC encapsulation on the logical interface. You also need to configure an encapsulation on the physical interface. The physical interface encapsulation does not have to be a CCC encapsulation. However, it should match the logical interface encapsulation. For example, if you configure an ATM CCC encapsulation type on the logical interface, you should configure a compatible ATM encapsulation on the physical interface.

To configure the CCC encapsulation type, include the **encapsulation-type** statement:

encapsulation-type *ccc-encapsulation-type*;

To configure the CCC encapsulation type on the physical interface, include this statement at the following hierarchy levels:

- **[edit interfaces** *interface-name***]**
- **[edit logical-systems** *logical-system-name* **interfaces** *interface-name***]**

To configure the CCC encapsulation type on the logical interface, include this statement at the following hierarchy levels:

- **[edit interfaces** *interface-name* **unit** *logical-unit-number***]**
- **[edit logical-systems** *logical-system-name* **interfaces** *interface-name* **unit** *logical-unit-number***]**

You configure the encapsulation type at the **[edit interfaces]** hierarchy level differently from the **[edit routing-instances]** hierarchy level. For example, you specify the encapsulation as **frame-relay** at the **[edit routing-instances]** hierarchy level and as **frame-relay-ccc** at the **[edit interfaces]** hierarchy level.

You can run both standard Frame Relay and CCC Frame Relay on the same device. If you specify Frame Relay encapsulation (**frame-relay-ccc**) for the interface, you should also configure the encapsulation at the **[edit interfaces** *interface name* **unit** *unit-number***]** hierarchy level as **frame-relay-ccc**. Otherwise, the logical interface unit defaults to standard Frame Relay.

For more information about how to configure interfaces and interface encapsulations, see the *Junos OS Network Interfaces Configuration Guide*.

Configuring TCC Encapsulation for Layer 2 VPNs and Layer 2 Circuits

Also known as Layer 2.5 VPNs, the translation cross-connect (TCC) encapsulation types allow you to configure different encapsulation types at the ingress and egress of a Layer 2 VPN or the ingress and egress of a Layer 2 circuit. For example, a CE router at the ingress of a Layer 2 VPN path can send traffic in a Frame Relay encapsulation. A CE router at the egress of that path can receive the traffic in an ATM encapsulation.

For information about how to configure encapsulations for Layer 2 circuits, see “Configuring the Interface Encapsulation Type for Layer 2 Circuits” on page 648.

The configuration for TCC encapsulation types is similar to the configuration for CCC encapsulation types. For Layer 2 VPNs, you specify a TCC encapsulation type for each PE-router-to-CE-router interface. The encapsulation type configured for the interface should match the encapsulation type configured under the routing instance. For information about how to configure the encapsulation type under the routing instance, see “Configuring the Encapsulation Type” on page 82.

You need to configure the TCC encapsulation on both the physical and logical interfaces. To configure the TCC encapsulation type, include the **encapsulation-type** statement:

encapsulation-type *tcc-encapsulation-type*;

To configure the TCC encapsulation type on the physical interface, include this statement at the following hierarchy levels:

- **[edit interfaces *interface-name*]**
- **[edit logical-systems *logical-system-name* interfaces *interface-name*]**

To configure the TCC encapsulation type on the logical interface, include this statement at the following hierarchy levels:

- **[edit interfaces *interface-name* unit *logical-unit-number*]**
- **[edit logical-systems *logical-system-name* interfaces *interface-name* unit *logical-unit-number*]**

You configure the encapsulation type at the **[edit interfaces]** hierarchy level differently than at the **[edit routing-instances]** hierarchy level. For example, you specify the encapsulation as **frame-relay** at the **[edit routing-instances]** hierarchy level and as **frame-relay-tcc** at the **[edit interfaces]** hierarchy level.

For Layer 2.5 VPNs employing an Ethernet interface as the TCC router, you can configure an Ethernet TCC or an extended VLAN TCC.

To configure an Ethernet TCC or an extended VLAN TCC, include the **proxy** and **remote** statements:

```
proxy inet-address;
remote (inet-address | mac-address);
```

You can include these statements at the following hierarchy levels:

- **[edit interfaces *interface-name* unit *logical-unit-number* family tcc]**
- **[edit logical-interfaces *logical-interface-name* interfaces *interface-name* unit *logical-unit-number* family tcc]**

The **proxy inet-address** address statement defines the IP address for which the TCC router is acting as proxy.

The **remote (inet-address | mac-address)** statement defines the location of the remote router.

Ethernet TCC is supported on interfaces that carry IP version 4 (IPv4) traffic only. Ethernet TCC encapsulation is supported on 1-port Gigabit Ethernet, 2-port Gigabit Ethernet, 4-port Gigabit Ethernet, and 4-port Fast Ethernet Physical Interface Cards (PICs) only.

For more information about how to configure interfaces and interface encapsulations, see the *Junos OS Network Interfaces Configuration Guide*.

Configuring Traffic Policing in Layer 2 VPNs

You can use policing to control the amount of traffic flowing over the interfaces servicing a Layer 2 VPN. If policing is disabled on an interface, all the available bandwidth on a Layer 2 VPN tunnel can be used by a single CCC or TCC interface.

For more information about the **policer** statement, see the *Junos OS Policy Framework Configuration Guide*.

To enable Layer 2 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]**

For information about how to configure the encapsulation type, see "Configuring the Encapsulation Type" on page 82.

Disabling the Control Word for Layer 2 VPNs

A 4-byte control word provides support for the emulated VC encapsulation for Layer 2 VPNs. This control word is added between the Layer 2 protocol data unit (PDU) being transported and the VC label that is used for demultiplexing. Various networking formats (ATM, Frame Relay, Ethernet, and so on) use the control word in a variety of ways.

On networks with equipment that does not support the control word, you can disable it by including the **no-control-word** statement:

```
no-control-word;
```

You can include this statement at the following hierarchy levels:

- **[edit routing-instances** *routing-instance-name* **protocols l2vpn]**
- **[edit logical-systems** *logical-system-name* **routing-instances** *routing-instance-name* **protocols l2vpn]**

For more information about configuring the control word, see "Configuring the Control Word for Layer 2 Circuits" on page 642 and the *Junos Feature Guide*.

NOTE: Use the **no-control-word** statement to disable the control word when the topology uses generic routing encapsulation (GRE) as the connection mechanism between PEs, and one of the PEs is an M Series router.

CHAPTER 7

Introduction to Layer 2 VPN Configuration Example

This chapter provides examples of Layer 2 virtual private networks(VPNs)..

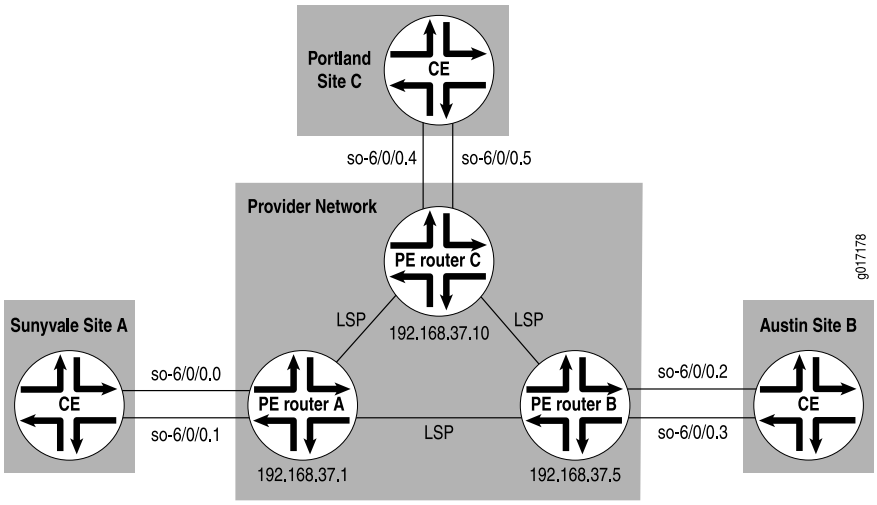
Layer 2 VPN Configuration Example

The following sections explain how to configure Layer 2 VPN functionality on the provider edge (PE) routers connected to each site:

Simple Full-Mesh Layer 2 VPN Overview

In the sections that follow, you configure a simple full-mesh Layer 2 VPN spanning three sites: Sunnyvale, Austin, and Portland. Each site connects to a PE router. The customer edge (CE) routers at each site use Frame Relay to carry Layer 2 traffic to the PE routers. Since this example uses a full-mesh topology between all three sites, each site requires two logical interfaces (one for each of the other CE routers), although only one physical link is needed to connect each PE router to each CE router. Figure 7 on page 90 illustrates the topology of this Layer 2 VPN.

Figure 7: Example of a Simple Full-Mesh Layer 2 VPN Topology



Enabling an IGP on the PE Routers

To allow the PE routers to exchange routing information among themselves, you must configure an interior gateway protocol (IGP) or static routes on these routers. You configure the IGP on the master instance of the routing protocol process (**rpd**) (that is, at the **[edit protocols]** hierarchy level), not within the Layer 2 VPN routing instance (that is, not at the **[edit routing-instances]** hierarchy level). Turn on traffic engineering on the IGP.

You configure the IGP in the standard way. This example does not include this portion of the configuration.

Configuring MPLS LSP Tunnels Between the PE Routers

In this configuration example, RSVP is used for MPLS signaling. Therefore, in addition to configuring RSVP, you must create an MPLS label-switched path (LSP) to tunnel the VPN traffic.

On Router A, enable RSVP and configure one end of the MPLS LSP tunnel to Router B. When configuring the MPLS LSP, include all interfaces using the **interface all** statement.

```
[edit]
protocols {
   rsvp {
      interface all;
   }
   mpls {
      interface all;
      label-switched-path RouterA-to-RouterB {
         to 192.168.37.5;
         primary Path-to-RouterB;
      }
      label-switched-path RouterA-to-RouterC {
         to 192.168.37.10;
         primary Path-to-RouterC;
```

```
    }
  }
}
```

On Router B, enable RSVP and configure the other end of the MPLS LSP tunnel. Again, configure the interfaces by using the **interface all** statement.

```
[edit]
protocols {
  rsvp {
    interface all;
  }
  mpls {
    interface all;
    label-switched-path RouterB-to-RouterA {
      to 192.168.37.1;
      primary Path-to-RouterA;
    }
    label-switched-path RouterB-to-RouterC {
      to 192.168.37.10;
      primary Path-to-RouterC;
    }
  }
}
```

On Router C, enable RSVP and configure the other end of the MPLS LSP tunnel. Again, configure all interfaces using the **interface all** statement.

```
[edit]
protocols {
  rsvp {
    interface all;
  }
  mpls {
    interface all;
    label-switched-path RouterC-to-RouterA {
      to 192.168.37.1;
      primary Path-to-RouterA;
    }
    label-switched-path RouterC-to-RouterB {
      to 192.168.37.5;
      primary Path-to-RouterB;
    }
  }
}
```

Configuring IBGP on the PE Routers

On the PE routers, configure an IBGP session with the following parameters:

- Layer 2 VPN—To indicate that the IBGP session is for a Layer 2 VPN, include the **family l2vpn** statement.
- Local address—The IP address in the **local-address** statement is the same as the address configured in the **to** statement at the **[edit protocols mpls label-switched-path**

lsp-path-name] hierarchy level on the remote PE router. The IBGP session for Layer 2 VPNs runs through this address.

- Neighbor address—Include the **neighbor** statement, specifying the IP address of the neighboring PE router.

On Router A, configure IBGP:

```
[edit]
protocols {
  bgp {
    import match-all;
    export match-all;
    group pe-pe {
      type internal;
      neighbor 192.168.37.5 {
        local-address 192.168.37.1;
        family l2vpn {
          signaling;
        }
      }
      neighbor 192.168.37.10 {
        local-address 192.168.37.1;
        family l2vpn {
          signaling;
        }
      }
    }
  }
}
```

On Router B, configure IBGP:

```
[edit]
protocols {
  bgp {
    local-address 192.168.37.5;
    import match-all;
    export match-all;
    group pe-pe {
      type internal;
      neighbor 192.168.37.1 {
        local-address 192.168.37.5;
        family l2vpn {
          signaling;
        }
      }
      neighbor 192.168.37.10 {
        local-address 192.168.37.5;
        family l2vpn {
          signaling;
        }
      }
    }
  }
}
```

On Router C, configure IBGP:

```
[edit]
protocols {
    bgp {
        local-address 192.168.37.10;
        import match-all;
        export match-all;
        group pe-pe {
            type internal;
            neighbor 192.168.37.1 {
                local-address 192.168.37.10;
                family l2vpn {
                    signaling;
                }
            }
            neighbor 192.168.37.5 {
                local-address 192.168.37.10;
                family l2vpn {
                    signaling;
                }
            }
        }
    }
}
```

Configuring Routing Instances for Layer 2 VPNs on the PE Routers

The three PE routers service the Layer 2 VPN, so you need to configure a routing instance on each router. For the VPN, you must define the following in each 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 **l2vpn**, which configures the router to run a Layer 2 VPN.
- Interfaces connected to the CE routers.
- VPN routing and forwarding (VRF) import and export policies, which must be the same on each PE router that services the same VPN and are used to control the network topology. Unless the import policy contains only a **then reject** statement, it must include a reference to a community. Otherwise, when you attempt to commit the configuration, the commit operation fails.

On Router A, configure the following routing instance for the Layer 2 VPN:

```
[edit]
routing-instances {
    VPN-Sunnyvale-Portland-Austin {
        instance-type l2vpn;
        interface so-6/0/0.0;
        interface so-6/0/0.1;
        route-distinguisher 100:1;
        vrf-import vpn-SPA-import;
        vrf-export vpn-SPA-export;
        protocols {
```

```
        l2vpn {
          encapsulation-type frame-relay;
          site Sunnyvale {
            site-identifier 1;
            interface so-6/0/0.0 {
              remote-site-id 2;
            }
            interface so-6/0/0.1 {
              remote-site-id 3;
            }
          }
        }
      }
    }
  }
```

On Router B, configure the following routing instance for the Layer 2 VPN:

```
[edit]
routing-instances {
  VPN-Sunnyvale-Portland-Austin {
    instance-type l2vpn;
    interface so-6/0/0.2;
    interface so-6/0/0.3;
    route-distinguisher 100:1;
    vrf-import vpn-SPA-import;
    vrf-export vpn-SPA-export;
    protocols {
      l2vpn {
        encapsulation-type frame-relay;
        site Austin {
          site-identifier 2;
          interface so-6/0/0.2 {
            remote-site-id 1;
          }
          interface so-6/0/0.3 {
            remote-site-id 3;
          }
        }
      }
    }
  }
}
```

On Router C, configure the following routing instance for the Layer 2 VPN:

```
[edit]
routing-instances {
  VPN-Sunnyvale-Portland-Austin {
    instance-type l2vpn;
    interface so-6/0/0.4;
    interface so-6/0/0.5;
    route-distinguisher 100:1;
    vrf-import vpn-SPA-import;
    vrf-export vpn-SPA-export;
    protocols {
      l2vpn {
```

```
            encapsulation-type frame-relay;
            site Portland {
               site-identifier 3;
               interface so-6/0/0.4 {
                  remote-site-id 1;
               }
               interface so-6/0/0.5 {
                  remote-site-id 2;
               }
            }
         }
      }
   }
}
```

Configuring CCC Encapsulation on the Interfaces

You need to specify a circuit cross-connect (CCC) encapsulation type for each PE-router-to-CE-router interface running in the Layer 2 VPN. This encapsulation type should match the encapsulation type configured under the routing instance.

Configure the following CCC encapsulation types for the interfaces on Router A:

```
[edit]
interfaces so-6/0/0 {
   encapsulation frame-relay-ccc;
   unit 0 {
      encapsulation frame-relay-ccc;
   }
}
interfaces so-6/0/0 {
   encapsulation frame-relay-ccc;
   unit 1 {
      encapsulation frame-relay-ccc;
   }
}
```

Configure the following CCC encapsulation types for the interfaces on Router B:

```
[edit]
interfaces so-6/0/0 {
   encapsulation frame-relay-ccc;
   unit 2 {
      encapsulation frame-relay-ccc;
   }
}
interfaces so-6/0/0 {
   encapsulation frame-relay-ccc;
   unit 3 {
      encapsulation frame-relay-ccc;
   }
}
```

Configure the following CCC encapsulation types for the interfaces on Router C:

```
[edit]
interface so-6/0/0 {
```

```
    encapsulation frame-relay-ccc;
    unit 4 {
      encapsulation frame-relay-ccc;
    }
  }
  interface so-6/0/0 {
    encapsulation frame-relay-ccc;
    unit 5 {
      encapsulation frame-relay-ccc;
    }
  }
```

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 the routers use to forward packets within the VPN.

NOTE: Use the community add *community-name* statement at the [edit policy-options policy-statement *policy-statement-name* term *term-name* then] hierarchy level to facilitate Layer 2 VPN VRF export policies.

On Router A, configure the following VPN import and export policies:

```
[edit]
policy-options {
  policy-statement match-all {
    term acceptable {
      then accept;
    }
  }
  policy-statement vpn-SPA-export {
    term a {
      then {
        community add SPA-com;
        accept;
      }
    }
    term b {
      then reject;
    }
  }
  policy-statement vpn-SPA-import {
    term a {
      from {
        protocol bgp;
        community SPA-com;
      }
      then accept;
    }
    term b {
      then reject;
    }
```

```
    }
    community SPA-com members target:69:100;
}
```

On Router B, configure the following VPN import and export policies:

```
[edit]
policy-options {
    policy-statement match-all {
        term acceptable {
            then accept;
        }
    }
    policy-statement vpn-SPA-import {
        term a {
            from {
                protocol bgp;
                community SPA-com;
            }
            then accept;
        }
        term b {
            then reject;
        }
    }
    policy-statement vpn-SPA-export {
        term a {
            then {
                community add SPA-com;
                accept;
            }
        }
        term b {
            then reject;
        }
    }
    community SPA-com members target:69:100;
}
```

On Router C, configure the following VPN import and export policies:

```
[edit]
policy-options {
    policy-statement match-all {
        term acceptable {
            then accept;
        }
    }
    policy-statement vpn-SPA-import {
        term a {
            from {
                protocol bgp;
                community SPA-com;
            }
            then accept;
        }
        term b {
```

```
        then reject;
      }
    }
    policy-statement vpn-SPA-export {
      term a {
        then {
          community add SPA-com;
          accept;
        }
      }
      term b {
        then reject;
      }
    }
    community SPA-com members target:69:100;
}
```

To apply the VPN policies on the routers, include the **vrf-export** and **vrf-import** statements when you configure the routing instance. 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 Router A, include the following statements:

```
[edit]
routing-instances {
  VPN-Sunnyvale-Portland-Austin {
    vrf-import vpn-SPA-import;
    vrf-export vpn-SPA-export;
  }
}
```

To apply the VPN policies on Router B, include the following statements:

```
[edit]
routing-instances {
  VPN-Sunnyvale-Portland-Austin {
    vrf-import vpn-SPA-import;
    vrf-export vpn-SPA-export;
  }
}
```

To apply the VPN policies on Router C, include the following statements:

```
[edit]
routing-instances {
  VPN-Sunnyvale-Portland-Austin {
    vrf-import vpn-SPA-import;
    vrf-export vpn-SPA-export;
  }
}
```

Layer 2 VPN Configuration Summarized by Router

For a summary of the configuration on each router in the examples in this chapter, see the following sections:

Summary for Router A (PE Router for Sunnyvale)

Routing Instance for Layer 2 VPN

```
[edit]
routing-instances {
   VPN-Sunnyvale-Portland-Austin {
      instance-type l2vpn;
      interface so-6/0/0.0;
      interface so-6/0/0.1;
      route-distinguisher 100:1;
      vrf-import vpn-SPA-import;
      vrf-export vpn-SPA-export;
      protocols {
         l2vpn {
            encapsulation-type frame-relay;
            site Sunnyvale {
               site-identifier 1;
               interface so-6/0/0.0 {
                  remote-site-id 2;
               }
               interface so-6/0/0.1 {
                  remote-site-id 3;
               }
            }
         }
      }
   }
}
```

Configure CCC Encapsulation Types for Interfaces

```
interfaces {
   interface so-6/0/0 {
      encapsulation frame-relay-ccc;
      unit 0 {
         encapsulation frame-relay-ccc;
      }
   }
   interface so-6/0/0 {
      encapsulation frame-relay-ccc;
      unit 1 {
         encapsulation frame-relay-ccc;
      }
   }
}
```

Master Protocol Instance

```
protocols {
}
```

Enable RSVP

```
rsvp {
  interface all;
}
```

Configure MPLS LSPs

```
mpls {
  label-switched-path RouterA-to-RouterB {
    to 192.168.37.5;
    primary Path-to-RouterB {
      cspf;
    }
  }
  label-switched-path RouterA-to-RouterC {
    to 192.168.37.10;
    primary Path-to-RouterC {
      cspf;
    }
  }
  interface all;
}
```

Configure IBGP

```
bgp {
  import match-all;
  export match-all;
  group pe-pe {
    type internal;
    neighbor 192.168.37.5 {
      local-address 192.168.37.1;
      family l2vpn {
        signaling;
      }
    }
    neighbor 192.168.37.10 {
      local-address 192.168.37.1;
      family l2vpn {
        signaling;
      }
    }
  }
}
```

Configure VPN Policy

```
policy-options {
  policy-statement match-all {
    term acceptable {
      then accept;
    }
  }
  policy-statement vpn-SPA-export {
    term a {
      then {
        community add SPA-com;
        accept;
      }
    }
    term b {
      then reject;
```

```
    }
  }
  policy-statement vpn-SPA-import {
    term a {
      from {
        protocol bgp;
        community SPA-com;
      }
      then accept;
    }
    term b {
      then reject;
    }
  }
  community SPA-com members target:69:100;
}
```

Summary for Router B (PE Router for Austin)

Routing Instance for VPN

```
[edit]
routing-instances {
  VPN-Sunnyvale-Portland-Austin {
    instance-type l2vpn;
    interface so-6/0/0.2;
    interface so-6/0/0.3;
    route-distinguisher 100:1;
    vrf-import vpn-SPA-import;
    vrf-export vpn-SPA-export;
  }
}
```

Configure Layer 2 VPN

```
protocols {
  l2vpn {
    encapsulation-type frame-relay;
    site Austin {
      site-identifier 2;
      interface so-6/0/0.2 {
        remote-site-id 1;
      }
      interface so-6/0/0.3 {
        remote-site-id 3;
      }
    }
  }
}
```

Configure CCC Encapsulation Types for Interfaces

```
[edit]
interfaces {
  interface so-6/0/0 {
    encapsulation frame-relay-ccc;
    unit 2 {
      encapsulation frame-relay-ccc;
    }
  }
  interface so-6/0/0 {
    encapsulation frame-relay-ccc;
```

```
    unit 3 {
      encapsulation frame-relay-ccc;
    }
  }
}
```

Master Protocol Instance

```
protocols {
}
```

Enable RSVP

```
rsvp {
  interface all;
}
```

Configure MPLS LSPs

```
mpls {
  label-switched-path RouterB-to-RouterA {
    to 192.168.37.1;
    primary Path-to-RouterA {
      cspf;
    }
  }
  label-switched-path RouterB-to-RouterC {
    to 192.168.37.10;
    primary Path-to-RouterC {
      cspf;
    }
  }
  interface all;
}
```

Configure IBGP

```
bgp {
  local-address 192.168.37.5;
  import match-all;
  export match-all;
  group pe-pe {
    type internal;
    neighbor 192.168.37.1 {
      local-address 192.168.37.5;
      family l2vpn {
        signaling;
      }
    }
    neighbor 192.168.37.10 {
      local-address 192.168.37.5;
      family l2vpn {
        signaling;
      }
    }
  }
}
```

Configure VPN Policy

```
policy-options {
  policy-statement match-all {
    term acceptable {
      then accept;
    }
```

```
    }
    policy-statement vpn-SPA-import {
      term a {
        from {
          protocol bgp;
          community SPA-com;
        }
        then accept;
      }
      term b {
        then reject;
      }
    }
    policy-statement vpn-SPA-export {
      term a {
        then {
          community add SPA-com;
          accept;
        }
      }
      term b {
        then reject;
      }
    }
    community SPA-com members target:69:100;
}
```

Summary for Router C (PE Router for Portland)

Routing Instance for VPN

```
[edit]
routing-instances {
    VPN-Sunnyvale-Portland-Austin {
        instance-type l2vpn;
        interface so-6/0/0.3;
        interface so-6/0/0.4;
        route-distinguisher 100:1;
        vrf-import vpn-SPA-import;
        vrf-export vpn-SPA-export;
    }
}
```

Configure Layer 2 VPN

```
protocols {
    l2vpn {
        encapsulation-type frame-relay;
        site Portland {
            site-identifier 3;
            interface so-6/0/0.4 {
                remote-site-id 1;
            }
            interface so-6/0/0.5 {
                remote-site-id 2;
            }
        }
    }
}
```

Configure CCC Encapsulation Types for Interfaces

```
[edit]
interfaces {
  interface so-6/0/0 {
    encapsulation frame-relay-ccc;
    unit 4 {
      encapsulation frame-relay-ccc;
    }
  }
  interface so-6/0/0 {
    encapsulation frame-relay-ccc;
    unit 5 {
      encapsulation frame-relay-ccc;
    }
  }
}
```

Master Protocol Instance

```
protocols {
}
```

Enable RSVP

```
rsvp {
  interface all;
}
```

Configure MPLS LSPs

```
mpls {
  label-switched-path RouterC-to-RouterA {
    to 192.168.37.1;
    primary Path-to-RouterA {
      cspf;
    }
  }
  label-switched-path RouterC-to-RouterB {
    to 192.168.37.5;
    primary Path-to-RouterB {
      cspf;
    }
  }
  interface all;
}
```

Configure IBGP

```
bgp {
  local-address 192.168.37.10;
  import match-all;
  export match-all;
  group pe-pe {
    type internal;
    neighbor 192.168.37.1 {
      local-address 192.168.37.10;
      family l2vpn {
        signaling;
      }
    }
    neighbor 192.168.37.5 {
      local-address 192.168.37.10;
      family l2vpn {
        signaling;
```

```
        }
      }
    }
  }
```

Configure VPN Policy

```
policy-options {
  policy-statement match-all {
    term acceptable {
      then accept;
    }
  }
  policy-statement vpn-SPA-import {
    term a {
      from {
        protocol bgp;
        community SPA-com;
      }
      then accept;
    }
    term b {
      then reject;
    }
  }
  policy-statement vpn-SPA-export {
    term a {
      then {
        community add SPA-com;
        accept;
      }
    }
    term b {
      then reject;
    }
  }
  community SPA-com members target:69:100;
}
```

Layer 2 VPN to Layer 2 VPN Connections

As the need to link different Layer 2 services to one another for expanded service offerings grows, Layer 2 MPLS VPN services are increasingly in demand. Junos OS enables you to terminate Layer 2 VPN into Layer 2 VPN (also known as Layer 2 VPN stitching) using the Layer 2 interworking (iw0) interface.

Another way to do this is to use a Tunnel Services PIC to loop packets out and back from the Packet Forwarding Engine (PFE), to link together Layer 2 networks. The Layer 2 interworking software interface avoids the need for the Tunnel Services PIC and overcomes the limitation of bandwidth constraints imposed by the Tunnel Services PIC.

Related Documentation

- Example: Interconnecting a Layer 2 VPN with a Layer 2 VPN on page 106

Example: Interconnecting a Layer 2 VPN with a Layer 2 VPN

This example provides a step-by-step procedure for interconnecting and verifying a Layer 2 VPN with a Layer 2 VPN. It contains the following sections:

Requirements

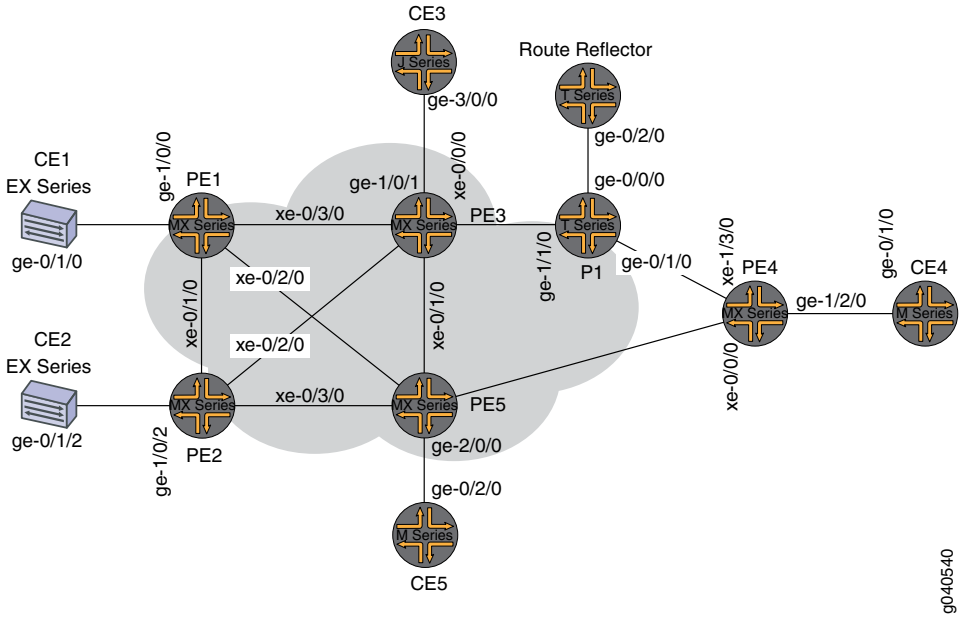
This example uses the following hardware and software components:

- Junos OS Release 9.3 or later
- 2 MX Series routers
- 2 M Series routers
- 1 T Series router
- 1 EX Series router
- 1 J Series router

Overview and Topology

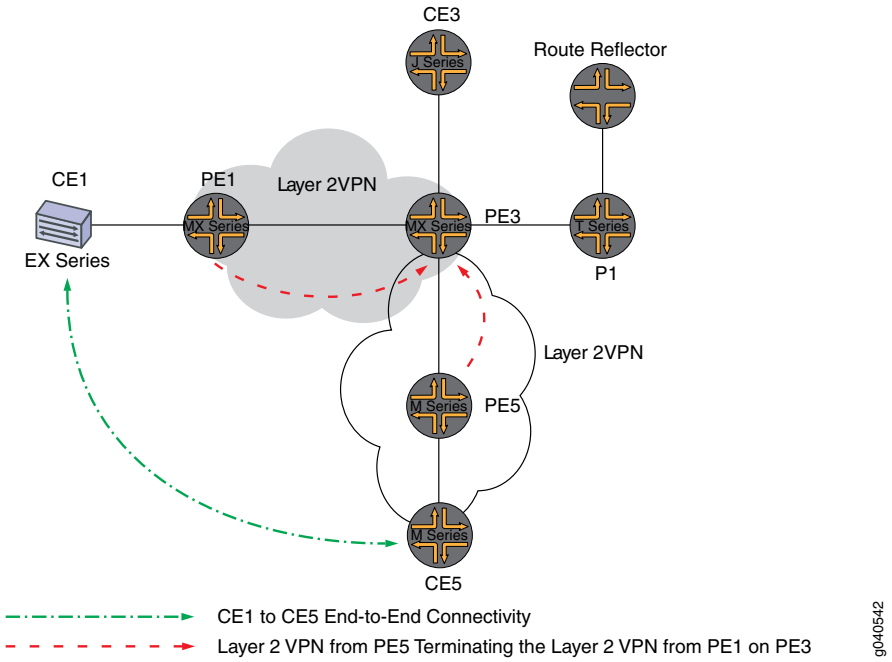
The physical topology of the Layer 2 VPN to Layer 2 VPN connection example is shown in Figure 8 on page 106.

Figure 8: Physical Topology of a Layer 2 VPN to Layer 2 VPN Connection



The logical topology of a Layer 2 VPN to Layer 2 VPN connection is shown in Figure 9 on page 107.

Figure 9: Logical Topology of a Layer 2 VPN to Layer 2 VPN Connection



Configuration

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:

- **CE1** identifies the customer edge 1 (CE1) 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 is organized in the following sections:

Configuring Protocols on the PE and P Routers

Step-by-Step Procedure

All of the PE routers and P routers are configured with OSPF as the IGP protocol. The MPLS, LDP, and BGP protocols are enabled on all of the interfaces except **fxp.0.** Core-facing interfaces are enabled with the MPLS address and inet address.

1. Configure all the PE and P routers with OSPF as the IGP. Enable the MPLS, LDP, and BGP protocols on all interfaces except **fxp.0**. The following configuration snippet shows the protocol configuration for Router PE1:

```
[edit]
protocols {
    mpls {
        interface all;
        interface fxp0.0 {
            disable;
        }
    }
    bgp {
        group RR {
            type internal;
            local-address 1.1.1.1;
            family l2vpn {
                signaling;
            }
            neighbor 7.7.7.7;
        }
    }
    ospf {
        traffic-engineering;
        area 0.0.0.0 {
            interface all;
            interface fxp0.0 {
                disable;
            }
        }
    }
    ldp {
        interface all;
        interface fxp0.0 {
            disable;
        }
    }
}
```

2. Configure the PE and P routers with OSPF as the IGP. Enable the MPLS, LDP, and BGP protocols on all interfaces except **fxp.0**. The following configuration snippet shows the protocol configuration for Router PE3:

```
[edit]
protocols {
    mpls {
        interface all;
        interface fxp0.0 {
            disable;
        }
```

```
    }
    bgp {
      group RR {
        type internal;
        local-address 3.3.3.3;
        family l2vpn {
          signaling;
        }
        neighbor 7.7.7.7;
      }
    }
    ospf {
      traffic-engineering;
      area 0.0.0.0 {
        interface all;
        interface fxp0.0 {
          disable;
        }
      }
    }
    ldp {
      interface all;
      interface fxp0.0 {
        disable;
      }
    }
}
```

Step-by-Step Procedure

Configuring the Layer 2 VPN Protocol and Interfaces

1. On Router PE1, configure the **ge-1/0/0** 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-1/0/0.0** 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 VPN domain.

   ```
   [edit interfaces]
   ge-1/0/0 {
     encapsulation ethernet-ccc;
     unit 0 {
       family ccc;
     }
   }
   lo0 {
     unit 0 {
       family inet {
         address 1.1.1.1/32;
       }
     }
   }
   ```

2. On Router PE1, configure the Layer 2 VPN protocols. Configure the remote site ID as 3. Site ID 3 represents Router PE3 (Hub-PE). To configure the Layer 2 VPN

protocols, include the **l2vpn** statement at the **[edit routing-instances *routing-instances-name* protocols]** hierarchy level. Layer 2 VPNs use BGP as the signaling protocol.

```
[edit routing-instances]
L2VPN {
    instance-type l2vpn;
    interface ge-1/0/0.0;
    route-distinguisher 65000:1;
    vrf-target target:65000:2;
    protocols {
        l2vpn {
            encapsulation-type ethernet;
            site CE1 {
                site-identifier 1;
                interface ge-1/0/0.0 {
                    remote-site-id 3;
                }
            }
        }
    }
}
```

3. On Router PE5, configure the **ge-2/0/0** interface encapsulation by including the **encapsulation** statement and specify the **ethernet-ccc** option. Configure the ge-1/0/0.0 logical interface family for circuit cross-connect functionality by including the **family** statement and specifying the **ccc** option.

```
[edit interfaces]
ge-2/0/0 {
    encapsulation ethernet-ccc;
    unit 0 {
        family ccc;
    }
}
lo0 {
    unit 0 {
        family inet {
            address 5.5.5.5/32;
        }
    }
}
```

4. On Router PE5, configure the Layer 2 VPN protocols by including the **l2vpn** statement at the **[edit routing-instances *routing-instances-name* protocols]** hierarchy level. Configure the remote site ID as **3**.

```
[edit routing-instances]
L2VPN {
    instance-type l2vpn;
    interface ge-2/0/0.0;
    route-distinguisher 65000:5;
    vrf-target target:65000:2;
    protocols {
        l2vpn {
            encapsulation-type ethernet;
```

```
        site CE5 {
          site-identifier 5;
          interface ge-2/0/0.0 {
            remote-site-id 3;
          }
        }
      }
    }
  }
```

5. On Router PE3, configure the **iw0** interface with two logical interfaces. To configure the **iw0** interface, include the **interfaces** statement and specify **iw0** as the interface name. For the unit 0 logical interface, include the **peer-unit** statement and specify the logical interface **unit 1** as the peer l interface. For the unit 1 logical interface, include the **peer-unit** statement and specify the logical interface **unit 0** as the peer interface.

```
[edit interfaces]
iw0 {
  unit 0 {
    encapsulation ethernet-ccc;
    peer-unit 1;
  }
  unit 1 {
    encapsulation ethernet-ccc;
    peer-unit 0;
  }
}
```

6. On Router PE3, configure the edge-facing **ge-1/0/1** interface encapsulation by including the **encapsulation** statement and specifying the **ethernet-ccc** option.

```
[edit interfaces]
ge-1/0/1 {
  encapsulation ethernet-ccc;
  unit 0 {
    family ccc;
  }
}
```

7. On Router PE3, configure the logical loopback interface. The loopback interface is used to establish the targeted LDP sessions to Routers PE1 and PE5.

```
[edit interfaces]
lo0 {
  unit 0 {
    family inet {
      address 3.3.3.3/32;
    }
  }
}
```

8. On Router PE3, enable the Layer 2 interworking protocol. To enable the Layer 2 interworking protocol, include the **l2iw** statement at the **[edit protocols]** hierarchy level.

```
[edit protocols]
```

```
l2iw;
```

9. On Router PE3, configure two Layer 2 VPN routing instances to terminate the Layer 2 VPN virtual circuits from Routers PE1 and PE5, as shown.

```
[edit routing-instances]
L2VPN-PE1 {
   instance-type l2vpn;
   interface iw0.0;
   route-distinguisher 65000:3;
   vrf-target target:65000:2;
   protocols {
      l2vpn {
         encapsulation-type ethernet;
         site CE3 {
            site-identifier 3;
            interface iw0.0 {
               remote-site-id 1;
            }
         }
      }
   }
}
L2VPN-PE5 {
   instance-type l2vpn;
   interface iw0.1;
   route-distinguisher 65000:33;
   vrf-target target:65000:2;
   protocols {
      l2vpn {
         encapsulation-type ethernet;
         site CE3 {
            site-identifier 3;
            interface iw0.1 {
               remote-site-id 5;
            }
         }
      }
   }
}
```

Verifying the Layer 2 VPN to Layer 2 VPN Connection on Router PE3

Step-by-Step Procedure

1. BGP is used for control plane signaling in a Layer 2 VPN. On Router PE1, use the **show bgp** command to verify that the BGP control plane for the Layer 2 VPN, has established a neighbor relationship with the router reflector that has IP address **7.7.7.7**.

   Three Layer 2 VPN routes are received from the route reflector from each PE router in the topology.

```
user@PE1> show bgp summary

Groups: 1 Peers: 1 Down peers: 0
Table          Tot Paths  Act Paths Suppressed    History Damp State    Pending
bgp.l2vpn.0            3          3          0          0          0          0
Peer                     AS      InPkt     OutPkt    OutQ   Flaps Last Up/Dwn
State|#Active/Received/Accepted/Damped...
7.7.7.7               65000        190        192       0       0     1:24:40 Establ
  bgp.l2vpn.0: 3/3/3/0
  L2VPN.l2vpn.0: 3/3/3/0
```

2. On Router PE1, use the **show route** command to verify that the BGP Layer 2 VPN routes are stored in the **L2VPN.l2vpn.0** routing table for each PE router.

```
user@PE1> show route table L2VPN.l2vpn.0

L2VPN.l2vpn.0: 4 destinations, 4 routes (4 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both
65000:1:1:3/96
                   *[L2VPN/170/-101] 01:31:53, metric2 1
                      Indirect
65000:3:3:1/96
                   *[BGP/170] 01:24:58, localpref 100, from 7.7.7.7
                      AS path: I
                    > to 10.10.1.2 via xe-0/3/0.0
65000:5:5:3/96
                   *[BGP/170] 01:24:58, localpref 100, from 7.7.7.7
                      AS path: I
                    > to 10.10.3.2 via xe-0/2/0.0
65000:33:3:5/96
                   *[BGP/170] 01:24:58, localpref 100, from 7.7.7.7
                      AS path: I
                    > to 10.10.1.2 via xe-0/3/0.0
```

3. On Router PE1, use the **show ldp session** command to verify that targeted LDP sessions are established to the PE routers in the network and that the state is **Operational**.

```
user@PE1> show ldp session

  Address           State        Connection     Hold time
2.2.2.2             Operational  Open             24
3.3.3.3             Operational  Open             22
5.5.5.5             Operational  Open             28
```

4. On Router PE1, use the **show l2vpn connections** command to verify that the Layer 2 VPN to site 3 on Router PE3 (Hub-PE) is **Up**.

```
user@PE1> 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 availble
BK -- Backup connection          ST -- Standby connection
PF -- Profile parse failure      PB -- Profile busy

Legend for interface status
Up -- operational
Dn -- down

Instance: L2VPN
  Local site: CE1 (1)
    connection-site           Type  St     Time last up          # Up trans
    3                         rmt   Up     Jan  5 18:08:25 2010           1
      Remote PE: 3.3.3.3, Negotiated control-word: Yes (Null)
      Incoming label: 800000, Outgoing label: 800000
      Local interface: ge-1/0/0.0, Status: Up, Encapsulation: ETHERNET
    5                         rmt   OR
```

5. On Router PE1, use the **show route** command to verify that the **mpls.0** routing table is populated with the Layer 2 VPN routes used to forward the traffic using an LDP label. Notice that in this example, the router is pushing label **8000000**.

```
user@PE1> show route table mpls.0

[edit]
mpls.0: 13 destinations, 13 routes (13 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

0                  *[MPLS/0] 1w1d 11:36:44, metric 1
                      Receive
1                  *[MPLS/0] 1w1d 11:36:44, metric 1
                      Receive
2                  *[MPLS/0] 1w1d 11:36:44, metric 1
                      Receive
300432             *[LDP/9] 3d 04:25:02, metric 1
                    > to 10.10.2.2 via xe-0/1/0.0, Pop
300432(S=0)        *[LDP/9] 3d 04:25:02, metric 1
                    > to 10.10.2.2 via xe-0/1/0.0, Pop
300768             *[LDP/9] 3d 04:25:02, metric 1
                    > to 10.10.3.2 via xe-0/2/0.0, Pop
300768(S=0)        *[LDP/9] 3d 04:25:02, metric 1
                    > to 10.10.3.2 via xe-0/2/0.0, Pop
300912             *[LDP/9] 3d 04:25:02, metric 1
                    > to 10.10.3.2 via xe-0/2/0.0, Swap 299856
301264             *[LDP/9] 3d 04:24:58, metric 1
```

```
                    > to 10.10.1.2 via xe-0/3/0.0, Swap 308224
301312              *[LDP/9] 3d 04:25:01, metric 1
                    > to 10.10.1.2 via xe-0/3/0.0, Pop
301312(S=0)         *[LDP/9] 3d 04:25:01, metric 1
                    > to 10.10.1.2 via xe-0/3/0.0, Pop
800000      *[L2VPN/7] 01:25:28
        > via ge-1/0/0.0, Pop     Offset: 4
ge-1/0/0.0      *[L2VPN/7] 01:25:28, metric2 1
        > to 10.10.1.2 via xe-0/3/0.0, Push 800000 Offset: -4
```

Verifying the Layer 2 VPN to Layer 2 VPN Connection on Router PE3

Step-by-Step Procedure

1. On Router PE3, use the **show l2vpn connections** command to verify that the Layer 2 VPN connections from Router PE1 and Router PE5 are **Up** and are using the **iw0** interface.

```
user@PE3> show l2vpn connections

Instance: L2VPN-PE1
  Local site: CE3 (3)
    connection-site           Type  St     Time last up          # Up trans
    1                         rmt   Up     Jan  5 18:08:22 2010           1
      Remote PE: 1.1.1.1, Negotiated control-word: Yes (Null)
      Incoming label: 800000, Outgoing label: 800000
      Local interface: iw0.0, Status: Up, Encapsulation: ETHERNET
    5                         rmt   OR

Instance: L2VPN-PE5
  Local site: CE3 (3)
    connection-site           Type  St     Time last up          # Up trans
    1                         rmt   CN
    5                         rmt   Up     Jan  5 18:08:22 2010           1
      Remote PE: 5.5.5.5, Negotiated control-word: Yes (Null)
      Incoming label: 800002, Outgoing label: 800000
      Local interface: iw0.1, Status: Up, Encapsulation: ETHERNET
```

2. On Router PE3, use the **show ldp neighbor** command to verify that the targeted LDP session neighbor IP addresses are shown.

```
user@PE3> show ldp neighbor

Address            Interface          Label space ID         Hold time
1.1.1.1            lo0.0              1.1.1.1:0                44
2.2.2.2            lo0.0              2.2.2.2:0                42
4.4.4.4            lo0.0              4.4.4.4:0                31
5.5.5.5            lo0.0              5.5.5.5:0                44
```

3. On Router PE3, use the **show bgp summary** command to verify that the BGP control plane for the Layer 2 VPN, has established a neighbor relationship with the router reflector that has IP address **7.7.7.7.**

```
user@PE3> show bgp summary

Groups: 1 Peers: 1 Down peers: 0
Table          Tot Paths  Act Paths Suppressed    History Damp State    Pending
bgp.l2vpn.0            2          2          0          0          0          0
Peer                     AS      InPkt     OutPkt    OutQ   Flaps Last Up/Dwn
State|#Active/Received/Accepted/Damped...
7.7.7.7              65000      10092      10195       0       0  3d 4:23:27 Establ
  bgp.l2vpn.0: 2/2/2/0
```

```
L2VPN-PE1.l2vpn.0: 2/2/2/0
L2VPN-PE5.l2vpn.0: 2/2/2/0
```

4. On Router PE3, use the **show ldp session** command to verify that targeted LDP sessions are established to all of the PE routers in the network and that the state is **Operational**.

```
user@PE3> show ldp session

Address           State        Connection     Hold time
1.1.1.1             Operational  Open             24
2.2.2.2             Operational  Open             22
4.4.4.4             Operational  Open             20
5.5.5.5             Operational  Open             24
```

5. On Router PE3, use the **show route** command to verify that the **mpls.0** routing table is populated with the Layer 2 VPN routes used to forward the traffic using an LDP label. Notice that in this example, the router is swapping label **800000**. Also notice the two **iw0** interfaces that are used for the Layer 2 interworking routes.

```
user@PE3>show route table mpls.0

mpls.0: 16 destinations, 18 routes (16 active, 2 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

0                  *[MPLS/0] 1w1d 11:50:14, metric 1
                      Receive
1                  *[MPLS/0] 1w1d 11:50:14, metric 1
                      Receive
2                  *[MPLS/0] 1w1d 11:50:14, metric 1
                      Receive
308160             *[LDP/9] 3d 04:38:45, metric 1
                    > to 10.10.1.1 via xe-0/3/0.0, Pop
308160(S=0)        *[LDP/9] 3d 04:38:45, metric 1
                    > to 10.10.1.1 via xe-0/3/0.0, Pop
308176             *[LDP/9] 3d 04:38:44, metric 1
                    > to 10.10.6.2 via xe-0/1/0.0, Pop
308176(S=0)        *[LDP/9] 3d 04:38:44, metric 1
                    > to 10.10.6.2 via xe-0/1/0.0, Pop
308192             *[LDP/9] 00:07:18, metric 1
                    > to 10.10.20.1 via xe-0/0/0.0, Swap 601649
                      to 10.10.6.2 via xe-0/1/0.0, Swap 299856
308208             *[LDP/9] 3d 04:38:44, metric 1
                    > to 10.10.5.1 via xe-0/2/0.0, Pop
308208(S=0)        *[LDP/9] 3d 04:38:44, metric 1
                    > to 10.10.5.1 via xe-0/2/0.0, Pop
308224             *[LDP/9] 3d 04:38:42, metric 1
                    > to 10.10.20.1 via xe-0/0/0.0, Pop
308224(S=0)        *[LDP/9] 3d 04:38:42, metric 1
                    > to 10.10.20.1 via xe-0/0/0.0, Pop
800000       *[L2IW/6] 01:39:13, metric2 1
         > to 10.10.6.2 via xe-0/1/0.0, Swap 800000
         [L2VPN/7] 01:39:13
         > via iw0.0, Pop     Offset: 4
800002       *[L2IW/6] 01:39:13, metric2 1
         > to 10.10.1.1 via xe-0/3/0.0, Swap 800000
         [L2VPN/7] 01:39:13
         > via iw0.1, Pop     Offset: 4
iw0.0        *[L2VPN/7] 01:39:13, metric2 1
         > to 10.10.1.1 via xe-0/3/0.0, Push 800000 Offset: -4
```

```
iw0.1          *[L2VPN/7] 01:39:13, metric2 1
                > to 10.10.6.2 via xe-0/1/0.0, Push 800000 Offset: -4
```

Step-by-Step Procedure

Testing Layer 2 VPN to Layer 2 VPN Connectivity (CE1 to CE5)

1. On Router CE1, use the **ping** command to test connectivity to Router CE5. Notice that the response time is in milliseconds, confirming that the ping response is returned.

```
user@CE1>ping 40.40.40.11

PING 40.40.40.11 (40.40.40.11): 56 data bytes
64 bytes from 40.40.40.11: icmp_seq=1 ttl=64 time=22.425 ms
64 bytes from 40.40.40.11: icmp_seq=2 ttl=64 time=1.299 ms
64 bytes from 40.40.40.11: icmp_seq=3 ttl=64 time=1.032 ms
64 bytes from 40.40.40.11: icmp_seq=4 ttl=64 time=1.029 ms
```

2. On Router CE5, use the **ping** command to test connectivity to Router CE1. Notice that the response time is in milliseconds, confirming that the ping response is returned.

```
user@CE5>ping 40.40.40.1

PING 40.40.40.1 (40.40.40.1): 56 data bytes
64 bytes from 40.40.40.1: icmp_seq=0 ttl=64 time=1.077 ms
64 bytes from 40.40.40.1: icmp_seq=1 ttl=64 time=0.957 ms
64 bytes from 40.40.40.1: icmp_seq=2 ttl=64 time=1.057 ms 1.017 ms
```

Results

The configuration and verification of this example have been completed. The following section is for your reference.

The relevant sample configuration for Router PE1 follows.

Router PE1

```
chassis {
    dump-on-panic;
    fpc 1 {
        pic 3 {
            tunnel-services {
                bandwidth 1g;
            }
        }
    }
    network-services ethernet;
}
    interfaces {
        xe-0/1/0 {
            unit 0 {
                family inet {
                    address 10.10.2.1/30;
                }
                family mpls;
            }
        }
        xe-0/2/0 {
            unit 0 {
                family inet {
                    address 10.10.3.1/30;
```

```
        }
        family mpls;
      }
    }
    xe-0/3/0 {
      unit 0 {
        family inet {
          address 10.10.1.1/30;
        }
        family mpls;
      }
    }
    ge-1/0/0 {
      encapsulation ethernet-ccc;
      unit 0 {
        family ccc;
      }
    }
    lo0 {
      unit 0 {
        family inet {
          address 1.1.1.1/32;
        }
      }
    }
}
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 1.1.1.1;
        family l2vpn {
          signaling;
        }
        neighbor 7.7.7.7;
      }
    }
    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-1/0/0.0;
        route-distinguisher 65000:1;
        vrf-target target:65000:2;
        protocols {
          l2vpn {
            encapsulation-type ethernet;
            site CE1 {
              site-identifier 1;
              interface ge-1/0/0.0 {
                remote-site-id 3;
              }
            }
          }
        }
      }
    }
```

The relevant sample configuration for Router PE3 follows.

Router PE3

```
chassis {
  dump-on-panic;
  fpc 1 {
    pic 3 {
      tunnel-services {
        bandwidth 1g;
      }
    }
  }
  network-services ethernet;
}
interfaces {
  xe-0/0/0 {
    unit 0 {
      family inet {
        address 10.10.20.2/30;
      }
      family mpls;
    }
  }
  xe-0/1/0 {
    unit 0 {
      family inet {
        address 10.10.6.1/30;
      }
```

```
            family mpls;
        }
    }
    xe-0/2/0 {
        unit 0 {
            family inet {
                address 10.10.5.2/30;
            }
            family mpls;
        }
    }
    xe-0/3/0 {
        unit 0 {
            family inet {
                address 10.10.1.2/30;
            }
            family mpls;
        }
    }
    ge-1/0/1 {
        encapsulation ethernet-ccc;
        unit 0 {
            family ccc;
        }
    }
    iw0 {
        unit 0 {
            encapsulation ethernet-ccc;
            peer-unit 1;
        }
        unit 1 {
            encapsulation ethernet-ccc;
            peer-unit 0;
        }
    }
    lo0 {
        unit 0 {
            family inet {
                address 3.3.3.3/32;
            }
        }
    }
}
routing-options {
    static {
        route 172.0.0.0/8 next-hop 172.19.59.1;
    }
    autonomous-system 65000;
}
protocols {
    l2iw;
    mpls {
        interface all;
        interface fxp0.0 {
            disable;
        }
```

```
    }
    bgp {
      group RR {
        type internal;
        local-address 3.3.3.3;
        family l2vpn {
          signaling;
        }
        neighbor 7.7.7.7;
      }
    }
    ospf {
      area 0.0.0.0 {
        interface all;
        interface fxp0.0 {
          disable;
        }
      }
    }
    ldp {
      interface all;
      interface fxp0.0 {
        disable;
      }
    }
  }
  routing-instances {
    L2VPN-PE1 {
      instance-type l2vpn;
      interface iw0.0;
      route-distinguisher 65000:3;
      vrf-target target:65000:2;
      protocols {
        l2vpn {
          encapsulation-type ethernet;
          site CE3 {
            site-identifier 3;
            interface iw0.0 {
              remote-site-id 1;
            }
          }
        }
      }
    }
    L2VPN-PE5 {
      instance-type l2vpn;
      interface iw0.1;
      route-distinguisher 65000:33;
      vrf-target target:65000:2;
      protocols {
        l2vpn {
          encapsulation-type ethernet;
          site CE3 {
            site-identifier 3;
            interface iw0.1 {
              remote-site-id 5;
```

```
            }
          }
        }
      }
    }
  }
```

Related Documentation

- Layer 2 VPN Overview
- Layer 2 VPN Applications
- Using the Layer 2 Interworking Interface to Interconnect a Layer 2 VPN to a Layer 2 VPN

CHAPTER 8

Summary of Layer 2 VPN Configuration Statements

The following sections explain the major **routing-instances** configuration statements that apply specifically to Layer 2 virtual private networks (VPNs). The statements are organized alphabetically. Routing instances and the statements at the **[edit routing-instances *routing-instance-name* protocols]** hierarchy level are explained in the *Junos OS Routing Protocols Configuration Guide*.

control-channel

Syntax

```
control-channel {
  pwe3-control-word;
  pw-label-ttl-1;
  router-alert-label;
}
```

Hierarchy Level

[edit logical-systems *logical-system-name* routing-instances *routing-instance-name* protocols l2vpn oam],
[edit logical-systems *logical-system-name* routing-instances *routing-instance-name* protocols vpls oam],
[edit routing-instances *routing-instance-name* protocols l2vpn oam],
[edit routing-instances *routing-instance-name* protocols vpls oam]

Release Information

Statement introduced in Junos OS Release 10.0.

Description

Configure the Virtual Circuit Connection Verification (VCCV) BFD control channel. VCCV provides a control channel associated with a pseudowire. You can configure a number of different CV types for this control channel, based on the configuration of the pseudowire.

Options

pwe3-control-word—For BGP-based pseudowires that send OAM packets with a control word that has 0001b as the first nibble.

pw-label-ttl-1—For BGP-based pseudowires that send OAM packets with a router alert label.

router-alert-label—For BGP-based pseudowires that send OAM packets with the MPLS pseudowire label, time-to-live (TTL), set to 1.

Required Privilege Level

routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation

- Configuring BFD for VCCV for Layer 2 VPNs, Layer 2 Circuits, and VPLS on page 35

control-word

Syntax (control-word | no-control-word);

Hierarchy Level [edit logical-systems *logical-system-name* routing-instances *routing-instance-name* protocols l2vpn],
[edit routing-instances *routing-instance-name* protocols l2vpn]

Release Information Statement introduced before Junos OS Release 7.4.

Description Specify the control word. The control word is 4 bytes long and is inserted between the Layer 2 protocol data unit (PDU) being transported and the virtual connection (VC) label that is used for demultiplexing.

- **control-word**—Enables the use of the control word.
- **no-control-word**—Disables the use of the control word.

Default The control word is enabled by default. You can also configure the control word explicitly using the **control-word** statement.

Required Privilege Level routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation
- Disabling the Control Word for Layer 2 VPNs on page 87

description

Syntax description *text*;

Hierarchy Level [edit logical-systems *logical-system-name* routing-instances *routing-instance-name* protocols l2vpn site *site-name* interface *interface-name*],
[edit routing-instances *routing-instance-name* protocols l2vpn site *site-name* interface *interface-name*]

Release Information Statement introduced before Junos OS Release 7.4.
Statement introduced in Junos OS Release 11.1 for EX Series switches.

Description Describe the VPN or virtual private LAN service (VPLS) routing instance.

Options *text*—Provide a text description. If the text includes one or more spaces, enclose it 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 operation.

Required Privilege Level routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation
- Configuring the Site on page 80
- Configuring an MPLS-Based Layer 2 VPN (CLI Procedure)

encapsulation

See the following sections:

encapsulation (Logical Interface)

Syntax encapsulation (atm-ccc-cell-relay | atm-ccc-vc-mux | atm-cisco-nlpid | atm-mlppp-llc | atm-nlpid | atm-ppp-llc | atm-ppp-vc-mux | atm-snap | atm-tcc-snap | atm-tcc-vc-mux | atm-vc-mux | ether-over-atm-llc | ethernet-vpls-fr | ether-vpls-over-atm-llc | ethernet-vpls-ppp | ethernet | frame-relay-ccc | frame-relay-ppp | frame-relay-tcc | multilink-frame-relay-end-to-end | multilink-ppp | ppp-over-ether | ppp-over-ether-over-atm-llc | vlan-ccc | vlan-tcc | vlan-vpls);

Hierarchy Level [edit interfaces *interface-name* unit *logical-unit-number*],
[edit logical-systems *logical-system-name* interfaces *interface-name* unit *logical-unit-number*]

Release Information Statement introduced before Junos OS Release 7.4.

Description Logical link-layer encapsulation type.

Options

atm-ccc-cell-relay—Use Asynchronous Transfer Mode (ATM) cell relay encapsulation.

atm-ccc-vc-mux—Use ATM VC multiplex encapsulation on circuit cross-connect (CCC) circuits. When you use this encapsulation type, you can configure the family **ccc** only.

atm-cisco-nlpid—Use Cisco ATM Network Layer Protocol identifier (NLPID) encapsulation. When you use this encapsulation type, you can configure the family **inet** only.

atm-mlppp-llc—For ATM2 intelligent queuing (IQ) interfaces only, use Multilink Point-to-Point (MLPPP) over ATM adaptation layer 5 (AAL5) logical link control (LLC). For this encapsulation type, your routing platform must be equipped with a Link Services or Voice Services Physical Interface Card (PIC).

atm-nlpid—Use ATM NLPID encapsulation. When you use this encapsulation type, you can configure the family **inet** only.

atm-ppp-llc—For ATM2 IQ interfaces only, use Point-to-Point Protocol (PPP) over AAL5 logical link control (LLC) encapsulation.

atm-ppp-vc-mux—For ATM2 IQ interfaces only, use PPP over AAL5 multiplex encapsulation.

atm-snap—Use ATM Subnetwork Access Protocol (SNAP) encapsulation.

atm-tcc-snap—Use ATM SNAP encapsulation on translational cross-connect (TCC) circuits.

atm-tcc-vc-mux—Use ATM VC multiplex encapsulation on TCC circuits. When you use this encapsulation type, you can configure the family **tcc** only.

atm-vc-mux—Use ATM VC multiplex encapsulation. When you use this encapsulation type, you can configure the family **inet** only.

ether-over-atm-llc—For interfaces that carry IP version 4 (IPv4) traffic, use Ethernet over ATM LLC encapsulation. When you use this encapsulation type, you cannot configure multipoint interfaces.

ethernet-vpls-fr—Use in a VPLS setup when a CE device is connected to a PE device over a time division multiplexing (TDM) link. This encapsulation type enables the PE device to terminate the outer layer 2 Frame Relay connection, use the 802.1p bits inside the inner Ethernet header to classify the packets, look at the MAC address from the Ethernet header, and use it to forward the packet into a given VPLS instance.

ether-vpls-over-atm-llc—For ATM2 IQ interfaces only, use the Ethernet VPLS over ATM LLC encapsulation to bridge Ethernet interfaces and ATM interfaces over a VPLS routing instance (as described in RFC 2684, *Multiprotocol Encapsulation over ATM Adaptation Layer 5*). Packets from the ATM interfaces are converted to standard ENET2/802.3 encapsulated Ethernet frames with the frame check sequence (FCS) field removed.

ethernet-vpls-ppp—Use in a VPLS setup when a CE device is connected to a PE device over a time division multiplexing (TDM) link. This encapsulation type enables the PE device to terminate the outer layer 2 PPP connection, use the 802.1p bits inside the inner Ethernet header to classify the packets, look at the MAC address from the Ethernet header, and use it to forward the packet into a given VPLS instance.

ethernet—Use Ethernet II encapsulation (as described in RFC 894, *A Standard For The Transmission Of IP Datagrams Over Ethernet Networks*).

frame-relay-ccc—Use Frame Relay encapsulation on CCC circuits. When you use this encapsulation type, you can configure the family **ccc** only.

frame-relay-ppp—Use Frame Relay encapsulation on PPP circuits.

frame-relay-tcc—Use Frame Relay encapsulation on TCC circuits for connecting unlike media. When you use this encapsulation type, you can configure the family **tcc** only.

multilink-frame-relay-end-to-end—Use Multilink Frame Relay (MLFR) FRF.15 encapsulation. This encapsulation is used only on multilink, link services, and voice services interfaces and their constituent T1 or E1 interfaces.

multilink-ppp—Use MLPPP encapsulation. This encapsulation is used only on multilink, link services, and voice services interfaces and their constituent T1 or E1 interfaces.

ppp-over-ether—For underlying Ethernet interfaces on Juniper Networks J Series Services Routers only, use PPP over Ethernet encapsulation. When you use this encapsulation type, you cannot configure the interface address. Instead you configure the interface address on the PPP interface. For more information, see the *Junos OS Interfaces and Routing Configuration Guide*.

ppp-over-ether-over-atm-llc—For underlying ATM interfaces on J Series Services Routers only, use PPP over Ethernet over ATM LLC encapsulation. When you use this encapsulation type, you cannot configure the interface address. Instead you configure the interface address on the PPP interface. For more information, see the *Junos OS Interfaces and Routing Configuration Guide*.

vlan-ccc—Use Ethernet virtual LAN (VLAN) encapsulation on CCC circuits. When you use this encapsulation type, you can configure the family **ccc** only.

vlan-tcc—Use Ethernet VLAN encapsulation on TCC circuits. When you use this encapsulation type, you can configure the family **tcc** only.

vlan-vpls—Use Ethernet VLAN encapsulation on virtual private LAN service (VPLS) circuits.

Required Privilege Level

interface—To view this statement in the configuration.
interface-control—To add this statement to the configuration.

Related Documentation

- Configuring CCC Encapsulation for Layer 2 VPNs on page 84
- Configuring TCC Encapsulation for Layer 2 VPNs and Layer 2 Circuits on page 85

encapsulation (Physical Interface)

Syntax encapsulation (atm-ccc-cell-relay | atm-pvc | cisco-hdlc | cisco-hdlc-ccc | cisco-hdlc-tcc | ethernet-ccc | ethernet-over-atm | ethernet-tcc | ethernet-vpls | ethernet-vpls-fr | ethernet-vpls-ppp | extended-frame-relay-ccc | extended-frame-relay-tcc | extended-vlan-ccc | extended-vlan-tcc | extended-vlan-vpls | flexible-ethernet-services | flexible-frame-relay | frame-relay | frame-relay-ccc | frame-relay-port-ccc | frame-relay-tcc | multilink-frame-relay-uni-nni | ppp | ppp-ccc | ppp-tcc | vlan-ccc | vlan-vpls);

Hierarchy Level [edit interfaces *interface-name*],
[edit logical-systems *logical-system-name* interfaces *interface-name*]

Release Information Statement introduced before Junos OS Release 7.4.
Statement introduced in Junos OS Release 11.1 for EX Series switches.

Description Specify the physical link-layer encapsulation type. Not all encapsulation types are supported on the switches. See the switch CLI.

Default PPP encapsulation.

Options **atm-ccc-cell-relay**—Use ATM cell-relay encapsulation.

atm-pvc—Use ATM permanent virtual connection (PVC) encapsulation.

cisco-hdlc—Use Cisco-compatible HDLC framing.

cisco-hdlc-ccc—Use Cisco-compatible HDLC framing on CCC circuits.

cisco-hdlc-tcc—Use Cisco-compatible HDLC framing on TCC circuits for connecting unlike media.

ethernet-ccc—Use Ethernet CCC encapsulation on Ethernet interfaces that must accept packets carrying standard Tag Protocol ID (TPID) values. For example, Ethernet CCC encapsulation can be used to transparently transport any VLANs or other Ethernet frames entering a port across a Layer 2 circuit.

ethernet-over-atm—For interfaces that carry IPv4 traffic, use Ethernet over ATM encapsulation. When you use this encapsulation type, you cannot configure multipoint interfaces. As defined in RFC 1483 *Multiprotocol Encapsulation over ATM Adaptation Layer 5*, this encapsulation type allows ATM interfaces to connect to devices that support only bridged-mode protocol data units (BPDUs). The Junos OS does not completely support bridging, but accepts BPDU packets as a default gateway. If you use the router as an edge device, then the router acts as a default gateway. It accepts Ethernet LLC/SNAP frames with IP or Address Resolution Protocol (ARP) in the payload and drops the rest. For packets destined for the Ethernet LAN, a route lookup is done using the destination IP address. If the route lookup yields a full address match, the packet is encapsulated with an LLC/SNAP and media access control (MAC) header and forwarded to the ATM interface.

ethernet-tcc—For interfaces that carry IPv4 traffic, use Ethernet TCC encapsulation on interfaces that must accept packets carrying standard TPID values. Ethernet TCC is not currently supported on Fast Ethernet 48-port PICs.

ethernet-vpls—Use Ethernet VPLS encapsulation on Ethernet interfaces that have VPLS enabled and that must accept packets carrying standard TPID values.

ethernet-vpls-fr—Use in a VPLS setup when a CE device is connected to a PE device over a time division multiplexing (TDM) link. This encapsulation type enables the PE device to terminate the outer layer 2 Frame Relay connection, use the 802.1p bits inside the inner Ethernet header to classify the packets, look at the MAC address from the Ethernet header, and use it to forward the packet into a given VPLS instance.

ethernet-vpls-ppp—Use in a VPLS setup when a CE device is connected to a PE device over a time division multiplexing (TDM) link. This encapsulation type enables the PE device to terminate the outer layer 2 PPP connection, use the 802.1p bits inside the inner Ethernet header to classify the packets, look at the MAC address from the Ethernet header, and use it to forward the packet into a given VPLS instance.

extended-frame-relay-ccc—Use Frame Relay encapsulation on CCC circuits. This encapsulation type allows you to dedicate data link connection identifiers (DLCIs) 1 through 1022 to CCC.

extended-frame-relay-tcc—Use Frame Relay encapsulation on TCC circuits to connect unlike media. This encapsulation type allows you to dedicate DLCIs 1 through 1022 to TCC.

extended-vlan-ccc—Use extended VLAN encapsulation on CCC circuits with Gigabit Ethernet and 4-port Fast Ethernet interfaces that must accept packets carrying 802.1Q values.

extended-vlan-tcc—For interfaces that carry IPv4 traffic, use extended VLAN encapsulation on TCC circuits with Gigabit Ethernet interfaces on which you want to use 802.1Q tagging. Extended Ethernet TCC is not currently supported on Fast Ethernet 48-port PICs.

extended-vlan-vpls—Use extended VLAN VPLS encapsulation on Ethernet interfaces that have VLAN 802.1Q tagging and VPLS enabled and that must accept packets carrying TPIDs 0x8100, 0x9100, and 0x9901.

flexible-ethernet-services—For Gigabit Ethernet IQ interfaces and Gigabit Ethernet PICs with small form-factor pluggable transceivers (SFPs) only, use flexible Ethernet services encapsulation when you want to configure multiple per-unit Ethernet encapsulations. This encapsulation type allows you to configure any combination of route, TCC, CCC, and VPLS encapsulations on a single physical port. Aggregated Ethernet bundles cannot use this encapsulation type. If you configure flexible Ethernet services encapsulation on the physical interface, VLAN IDs from 1 through 511 are no longer reserved for normal VLANs.

flexible-frame-relay—For IQ interfaces only, use flexible Frame Relay encapsulation when you want to configure multiple per-unit Frame Relay encapsulations. This encapsulation type allows you to configure any combination of TCC, CCC, and standard Frame Relay encapsulations on a single physical port. Also, each logical interface can have any DLCI value from 1 through 1022.

frame-relay—Use Frame Relay encapsulation.

frame-relay-ccc—Use Frame Relay encapsulation or Frame Relay encapsulation on CCC circuits.

frame-relay-port-ccc—Use Frame Relay port CCC encapsulation to transparently carry all the DLCIs between two CE routers without explicitly configuring each DLCI on the two provider edge (PE) routers with Frame Relay transport. When you use this encapsulation type, you can configure the family **ccc** only.

frame-relay-tcc—Use Frame Relay encapsulation on TCC circuits to connect unlike media.

multilink-frame-relay-uni-nni—Use MLFR user-to-network interface (UNI) network-to-network interface (NNI) encapsulation. This encapsulation is used only on link services and voice services interfaces functioning as FRF.16 bundles and their constituent T1 or E1 interfaces.

ppp—Use serial PPP encapsulation.

ppp-ccc—Use serial PPP encapsulation on CCC circuits. When you use this encapsulation type, you can configure the family **ccc** only.

ppp-tcc—Use serial PPP encapsulation on TCC circuits for connecting unlike media. When you use this encapsulation type, you can configure the family **tcc** only.

vlan-ccc—Use Ethernet VLAN encapsulation on CCC circuits.

vlan-vpls—Use VLAN VPLS encapsulation on Ethernet interfaces with VLAN tagging and VPLS enabled. Interfaces with VLAN VPLS encapsulation accept packets carrying standard TPID values only.

Required Privilege Level

interface—To view this statement in the configuration.
interface-control—To add this statement to the configuration.

Related Documentation

- Configuring CCC Encapsulation for Layer 2 VPNs on page 84
- Configuring TCC Encapsulation for Layer 2 VPNs and Layer 2 Circuits on page 85
- Configuring an MPLS-Based Layer 2 VPN (CLI Procedure)

encapsulation-type

Syntax encapsulation-type (atm-aal5 | atm-cell | atm-cell-port-mode | atm-cell-vc-mode | atm-cell-vp-mode | cesop | cisco-hdlc | ethernet | ethernet-vlan | frame-relay | frame-relay-port-mode | interworking | ppp | satop-e1 | satop-e3 | satop-t1 | satop-t3);

Hierarchy Level [edit logical-systems *logical-system-name* routing-instances *routing-instance-name* protocols l2vpn],
[edit routing-instances *routing-instance-name* protocols l2vpn]

Release Information Statement introduced in Junos OS Release 9.2.
Statement introduced in Junos OS Release 11.1 for EX Series switches.

Description Specify the type of Layer 2 traffic originating from the CE router for the Layer 2 VPN. Not all encapsulation types are supported on the switches. See the switch CLI.

Options

atm-aal5—ATM Adaptation Layer (AAL/5)

atm-cell—ATM cell relay

atm-cell-port-mode—ATM cell relay port promiscuous mode

atm-cell-vc-mode—ATM VC cell relay nonpromiscuous mode

atm-cell-vp-mode—ATM virtual path (VP) cell relay promiscuous mode

cesop—CESOP-based Layer 2 VPN

cisco-hdlc—Cisco Systems–compatible HDLC

ethernet—Ethernet

ethernet-vlan—Ethernet VLAN

frame-relay—Frame Relay

frame-relay-port-mode—Frame Relay port mode

interworking—Layer 2.5 interworking VPN

ppp—PPP

satsop-e1—SATSOP-E1–based Layer 2 VPN

satsop-e3—SATSOP-E3–based Layer 2 VPN

satsop-t1—SATSOP-T1–based Layer 2 VPN

satsop-t3—SATSOP-T3–based Layer 2 VPN

Required Privilege Level routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation
- Configuring the Encapsulation Type on page 82
- Configuring an MPLS-Based Layer 2 VPN (CLI Procedure)

interface

Syntax

```
interface interface-name {
    description text;
    remote-site-id remote-site-id;
}
```

Hierarchy Level

[edit logical-systems *logical-system-name* routing-instances *routing-instance-name* protocols l2vpn],
[edit routing-instances *routing-instance-name* protocols l2vpn]

Release Information

Statement introduced before Junos OS Release 7.4.

Description

Configure an interface to handle traffic for a circuit configured for the Layer 2 VPN.

Options

interface-name—Name of the interface used for the Layer 2 VPN.

The remaining statements are explained separately.

Required Privilege Level

routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation
- Configuring the Site on page 80
- Configuring the Remote Site ID on page 80

l2vpn

Syntax

```
l2vpn {
  (control-word | no-control-word);
  encapsulation-type type;
  traceoptions {
    file filename <files number> <size size> <world-readable | no-world-readable>;
    flag flag <flag-modifier> <disable>;
  }
  site site-name {
    site-identifier identifier;
    site-preference preference-value {
      backup;
      primary;
    }
    interface interface-name {
      description text;
      remote-site-id remote-site-id;
    }
  }
}
```

Hierarchy Level

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

Release Information

Statement introduced before Junos OS Release 7.4.
Statement introduced in Junos OS Release 11.1 for EX Series switches.

Description

Enable a Layer 2 VPN routing instance on a PE router or switch.

The remaining statements are explained separately.

Required Privilege Level

routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation

- Configuring a Layer 2 VPN Routing Instance on page 79
- Configuring an MPLS-Based Layer 2 VPN (CLI Procedure)

no-control-word

See control-word

oam

Syntax

```
oam {
  ping-interval;
  bfd-liveness-detection {
    detection-time {
      threshold milliseconds;
    }
    minimum-interval milliseconds;
    minimum-receive-interval milliseconds;
    multiplier number;
    no-adaptation;
    transmit-interval {
      minimum-interval milliseconds;
      threshold milliseconds;
    }
    version bfd-protocol-version;
  }
  control-channel {
    pwe3–control-word;
    pseudowire-label-ttl-1;
    router-alert-label;
  }
}
```

Hierarchy Level

[edit routing-instances *routing-instance-name* protocols l2vpn],
[edit routing-instances *routing-instance-name* protocols vpls],
[edit routing-instances *routing-instance-name* protocols vpls neighbor *address*],
[edit protocols l2circuit neighbor *address* interface *interface-name*]

Release Information

Statement introduced in Junos OS Release 10.0.

Description

Allows you to configure bidirectional forwarding detection (BFD) and a control channel for a pseudowire, in addition to the corresponding operations and management functions to be used over that control channel. BFD provides a low resource fault detection mechanism for the continuous monitoring of the pseudowire data path and for detecting data plane failures. The **control-channel** statement is not applicable to Layer 2 circuit pseudowires.

Options

The **bfd-liveness-detection** statement and substatements are described in the *Junos OS Routing Protocols Configuration Guide*.

The other statements are explained separately.

Required Privilege Level

routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation

- Configuring BFD for VCCV for Layer 2 VPNs, Layer 2 Circuits, and VPLS on page 35

policer

Syntax

```
policer {
    input policer-template-name;
    output policer-template-name;
}
```

Hierarchy Level

[edit interfaces *interface-name* unit *logical-unit-number* family (ccc | inet | tcc)],
[edit logical-systems *logical-system-name* interfaces *interface-name* unit *logical-unit-number* family (ccc | inet | tcc)]

Release Information

Statement introduced before Junos OS Release 7.4.

Description

Use policing to control the amount of traffic flowing over the interfaces servicing a Layer 2 VPN.

Options

input *policer-template-name*—Name of one policer to evaluate when packets are received on the interface.

output *policer-template-name*—Name of one policer to evaluate when packets are transmitted on the interface.

Required Privilege Level

interface—To view this statement in the configuration.
interface-control—To add this statement to the configuration.

Related Documentation

- Configuring Traffic Policing in Layer 2 VPNs on page 86
- *Junos OS Policy Framework Configuration Guide*
- *Junos OS Network Interfaces Configuration Guide*

proxy

Syntax proxy inet-address *address*;

Hierarchy Level [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]

Release Information Statement introduced before Junos OS Release 7.4.

Description For Layer 2.5 VPNs using an Ethernet interface as the TCC router, configure the IP address for which the TCC router is proxying. Ethernet TCC is supported on interfaces that carry IPv4 traffic only. Ethernet TCC encapsulation is supported on 1-port Gigabit Ethernet, 2-port Gigabit Ethernet, 4-port Gigabit Ethernet, and 4-port Fast Ethernet PICs only. Ethernet TCC is not supported on the T640 routing node.

Options **inet-address *address***—IP address for which the TCC router is acting as a proxy.

Required Privilege Level interface—To view this statement in the configuration.
interface-control—To add this statement to the configuration.

Related Documentation
- Configuring TCC Encapsulation for Layer 2 VPNs and Layer 2 Circuits on page 85

remote

Syntax remote (inet-address | mac-address) *address*;

Hierarchy Level [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]

Release Information Statement introduced before Junos OS Release 7.4.

Description For Layer 2.5 VPNs employing an Ethernet interface as the TCC router, configure the location of the remote router. Ethernet TCC is supported on interfaces that carry IPv4 traffic only. Ethernet TCC encapsulation is supported on 1-port Gigabit Ethernet, 2-port Gigabit Ethernet, 4-port Gigabit Ethernet, and 4-port Fast Ethernet PICs only.

Options **inet-address*address***—The IP address of the remote site.

mac-address *address*—The MAC address of the remote site.

Required Privilege Level interface—To view this statement in the configuration.
interface-control—To add this statement to the configuration.

Related Documentation
- Configuring TCC Encapsulation for Layer 2 VPNs and Layer 2 Circuits on page 85

remote-site-id

Syntax remote-site-id *remote-site-ID*;

Hierarchy Level [edit logical-systems *logical-system-name* routing-instances *routing-instance-name* protocols l2vpn site *site-name* interface *interface-name*],
[edit routing-instances *routing-instance-name* protocols l2vpn site *site-name* interface *interface-name*]

Release Information Statement introduced before Junos OS Release 7.4.
Statement introduced in Junos OS Release 11.1 for EX Series switches.

Description Control the remote interface to which the interface should connect. If you do not explicitly configure the remote site ID, the order of the interfaces configured for the site determines the default value. This statement is optional.

Options ***remote-site-ID***—Identifier specifying the interface on the remote PE router the Layer 2 VPN routing instance connects to.

Required Privilege Level routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation
- Configuring the Remote Site ID on page 80
- Configuring an MPLS-Based Layer 2 VPN (CLI Procedure)

site

Syntax

```
site site-name {
  site-identifier identifier;
  site-preference preference-value {
    backup;
    primary;
  }
  interface interface-name {
    description text;
    remote-site-id remote-site-ID;
  }
}
```

Hierarchy Level

[edit logical-systems *logical-system-name* routing-instances *routing-instance-name* protocols l2vpn],
[edit routing-instances *routing-instance-name* protocols l2vpn]

Release Information

Statement introduced before Junos OS Release 7.4.
Statement introduced in Junos OS Release 11.1 for EX Series switches.

Description

Specify the site name, site identifier, and interfaces connecting to the site. Allows you to configure a remote site ID for remote sites.

Options

site-identifier *identifier*—Numerical identifier for the site used as a default reference for the remote site ID.

site-name—Name of the site.

The remaining statements are explained separately.

Required Privilege Level

routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation

- Configuring the Site on page 80
- Configuring an MPLS-Based Layer 2 VPN (CLI Procedure)

site-identifier

Syntax site-identifier *identifier*;

Hierarchy Level [edit logical-systems *logical-system-name* routing-instances *routing-instance-name* protocols l2vpn site *site-name*],
[edit routing-instances *routing-instance-name* protocols l2vpn site *site-name*]

Release Information Statement introduced before Junos OS Release 7.4.
Statement introduced in Junos OS Release 11.1 for EX Series switches.

Description Specify the numerical identifier for the local Layer 2 VPN site.

Options ***identifier***—The numerical identifier for the Layer 2 VPN site, which can be any number from 1 through 65,534.

Required Privilege Level routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation

- Configuring the Site on page 80
- Configuring an MPLS-Based Layer 2 VPN (CLI Procedure)

site-preference

Syntax

```
site-preference preference-value {
    backup;
    primary;
}
```

Hierarchy Level

[edit logical-systems *logical-system-name* routing-instances *routing-instance-name* protocols l2vpn site *site-name*],
[edit routing-instances *routing-instance-name* protocols l2vpn site *site-name*],

Release Information

Statement introduced in Junos OS Release 9.1.

Description

Specify the preference value advertised for a particular Layer 2 VPN site. The site preference value is encoded in the BGP local preference attribute. When a PE router receives multiple advertisements with the same VE identifier, the advertisement with the highest local preference value is preferred. You can use this statement to enable multihoming for Layer 2 VPNs.

Options

preference-value—Specify the preference value advertised for a Layer 2 VPN.

Range: 1 through 65,535

backup—Set the preference value to 1.

primary—Set the preference value to 65,535.

Required Privilege Level

routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation

- Configuring a Site Preference and Layer 2 VPN Multihoming on page 82

traceoptions

Syntax

```
traceoptions {
    file filename <files number> <size size> <world-readable | no-world-readable>;
    flag flag <flag-modifier> <disable>;
}
```

Hierarchy Level

[edit logical-systems *logical-system-name* routing-instances *routing-instance-name* protocols l2vpn],
[edit routing-instances *routing-instance-name* protocols l2vpn]

Release Information

Statement introduced before Junos OS Release 7.4.

Description

Trace traffic flowing through a Layer 2 VPN.

Options

disable—(Optional) Disable the tracing operation. You can use this option to disable a single operation when you have defined a broad group of tracing operations, such as **all**.

file *filename*—Name of the file to receive the output of the tracing operation. Enclose the name in quotation marks (" ").

files *number*—(Optional) Maximum number of trace files. When a trace file named ***trace-file*** reaches its maximum size, it is renamed ***trace-file*.0**, then ***trace-file*.1**, and so on, until the maximum number of trace files is reached. Then the oldest trace file is overwritten.

If you specify a maximum number of files, you also must specify a maximum file size with the **size** option.

Range: 2 through 1000 files

Default: 2 files

flag *flag*—Tracing operation to perform. To specify more than one tracing operation, include multiple **flag** statements.

- **all**—All Layer 2 VPN tracing options
- **connections**—Layer 2 connections (events and state changes)
- **error**—Error conditions
- **general**—General events
- **nlri**—Layer 2 advertisements received or sent by means of the BGP
- **normal**—Normal events
- **policy**—Policy processing
- **route**—Routing information
- **state**—State transitions
- **task**—Routing protocol task processing

- **timer**—Routing protocol timer processing
- **topology**—Layer 2 VPN topology changes caused by reconfiguration or advertisements received from other PE routers using BGP

flag-modifier—(Optional) Modifier for the tracing flag. You can specify the following modifier:

- **detail**—Provide detailed trace information
- **receive**—Trace received packets
- **send**—Trace transmitted packets

no-world-readable—(Optional) Prevents any user from reading the trace file.

size *size*—(Optional) Maximum size of each trace file, in kilobytes (KB), megabytes (MB), or gigabytes (GB). When a trace file named ***trace-file*** reaches this size, it is renamed ***trace-file*.0**. When ***trace-file*** again reaches its maximum size, ***trace-file*.0** is renamed ***trace-file*.1** and ***trace-file*** is renamed ***trace-file*.0**. This renaming scheme continues until the maximum number of trace files is reached. Then the oldest trace file is overwritten.

If you specify a maximum file size, you also must specify a maximum number of trace files with the **files** option.

Syntax: ***x*k** to specify kilobytes, ***x*m** to specify megabytes, or ***x*g** to specify gigabytes

Range: 10 KB through the maximum file size supported on your system

Default: 1 MB

world-readable—(Optional) Allow any user to read the trace file.

Default: The default is **no-world-readable**.

Required Privilege Level

routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation

- Tracing Layer 2 VPN Traffic and Operations on page 83

PART 3

Layer 3 VPNs

CHAPTER 9

Layer 3 VPN Overview

The Junos OS implements Layer 3 BGP/Multiprotocol Label Switching (BGP/MPLS) virtual private networks (VPNs) as defined in RFC 4364, *BGP/MPLS IP Virtual Private Networks (VPNs)*.

This chapter discusses the following topics that provide background information about Layer 3 VPNs:

Layer 3 VPN Introduction

In Junos OS, Layer 3 VPNs are based on RFC 4364. RFC 4364 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 4364 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 VPN identifier prefix 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.

Layer 3 VPN Platform Support

Layer 3 VPNs are supported on most combinations of Juniper Networks routing platforms and PICs capable of running the JUNOS Software.

MX Series routers configured to be in Ethernet services mode can support some of the Junos OS Layer 3 VPN features. For Layer 3 VPNs, Ethernet services mode supports configuring a loopback interface for a VPN routing and forwarding (VRF) instance. You can configure up to two VRF instances in Ethernet services mode. Each VRF instance can handle up to 10,000 routes. The **ping mpls l3vpn** operational mode command is also supported.

Layer 3 VPN Attributes

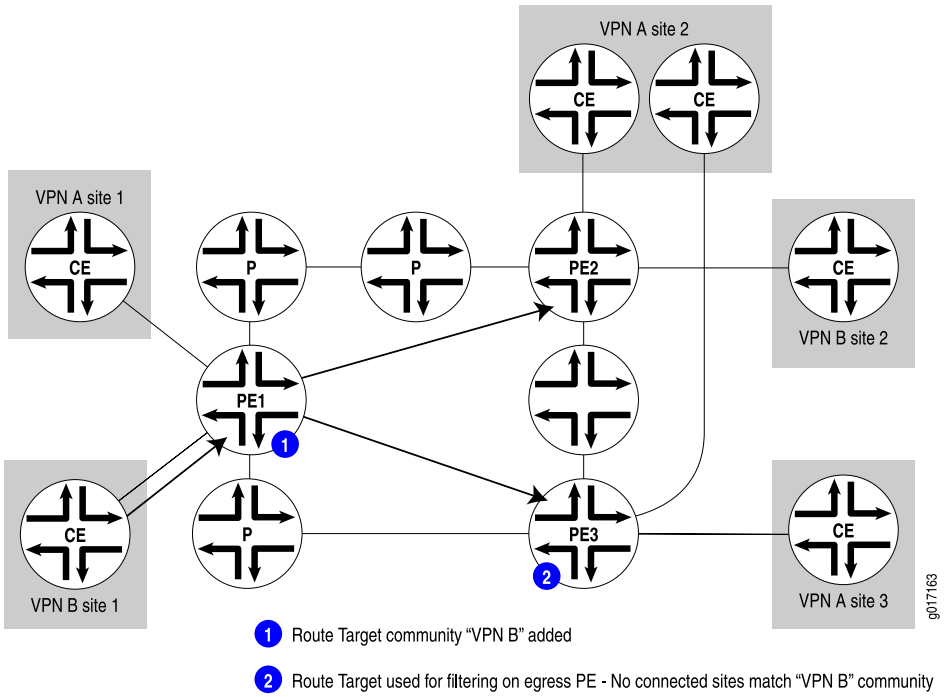
Route distribution within a VPN is controlled through BGP extended community attributes. RFC 4364 defines the following three attributes used by VPNs:

- 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 10 on page 149 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 10: VPN Attributes and Route Distribution

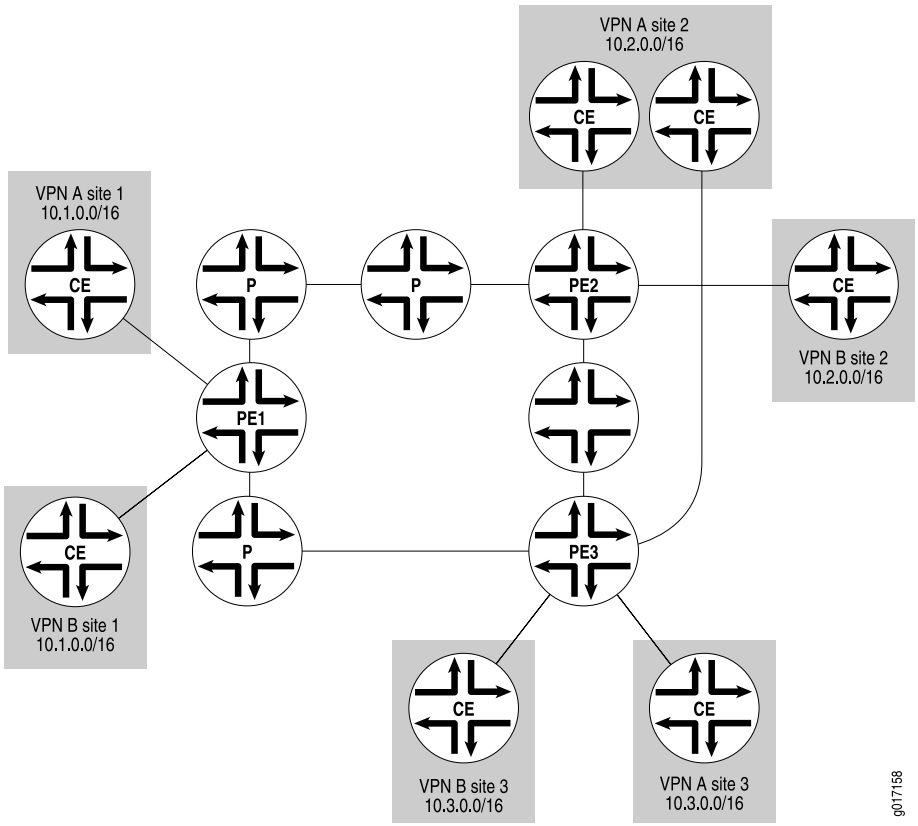


VPN-IPv4 Addresses and Route Distinguishers

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 11 on page 150 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 11: 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 11 on page 150 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**.

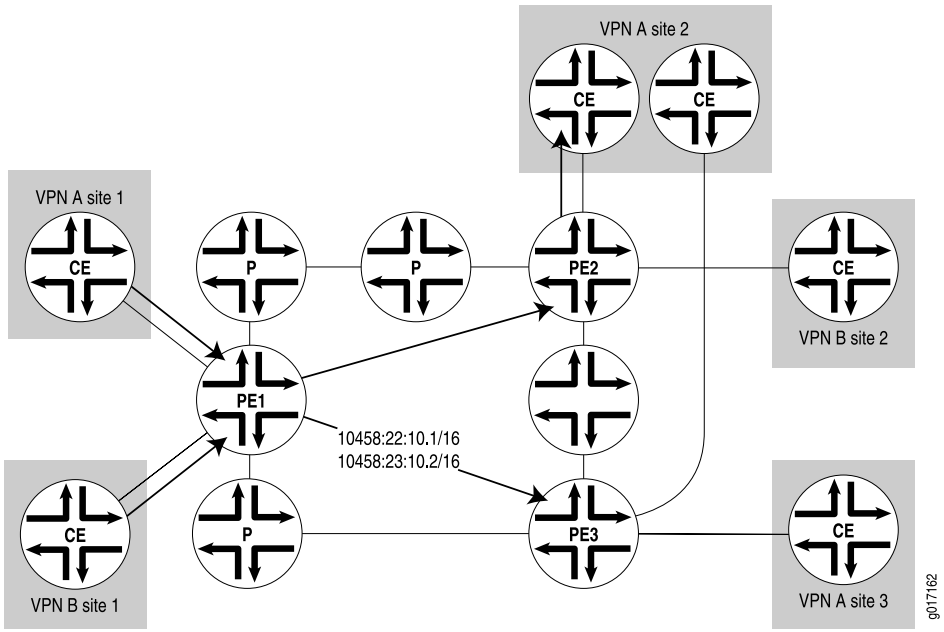
Route distinguishers are used only among PE routers to IPv4 addresses from different VPNs. The ingress PE router creates a route distinguisher and converts IPv4 routes received from CE routers into VPN-IPv4 addresses. The egress PE routers convert VPN-IPv4 routes into IPv4 routes before announcing them 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 12 on page 152 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 12: Route Distinguishers



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 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*.

For more information about IPv6, see the *Junos OS Routing Protocols Configuration Guide*.

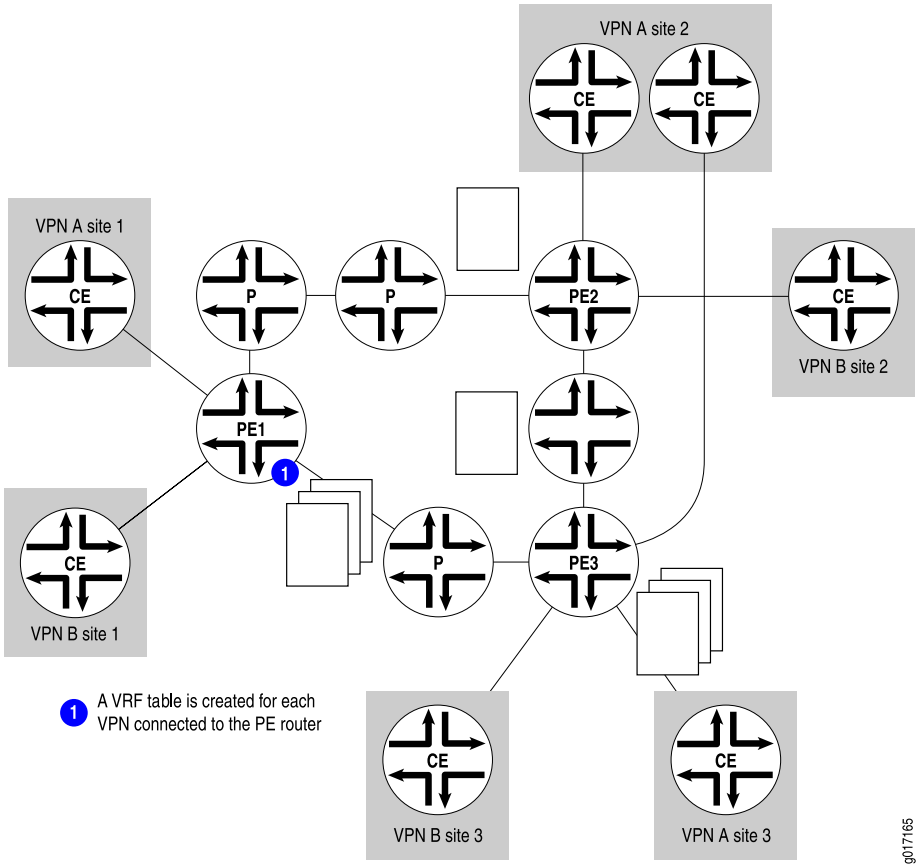
VPN Routing and Forwarding Tables

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 13 on page 153 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 13: 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 all VPN-IPv4 unicast routes received from other PE routers. (This table does not store routes received from directly connected CE routers.) This table is present only on PE 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 remote 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.

  Interior gateway protocol (IGP) shortcuts do not work in VPN environments and should not be configured. IGP shortcuts move routes in **inet.3** to **inet.0**. VPN IBGP (family **inet-vpn**) relies on next hops that are in the **inet.3** table; thus, IGP shortcuts are incompatible with VPNs.

- **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.

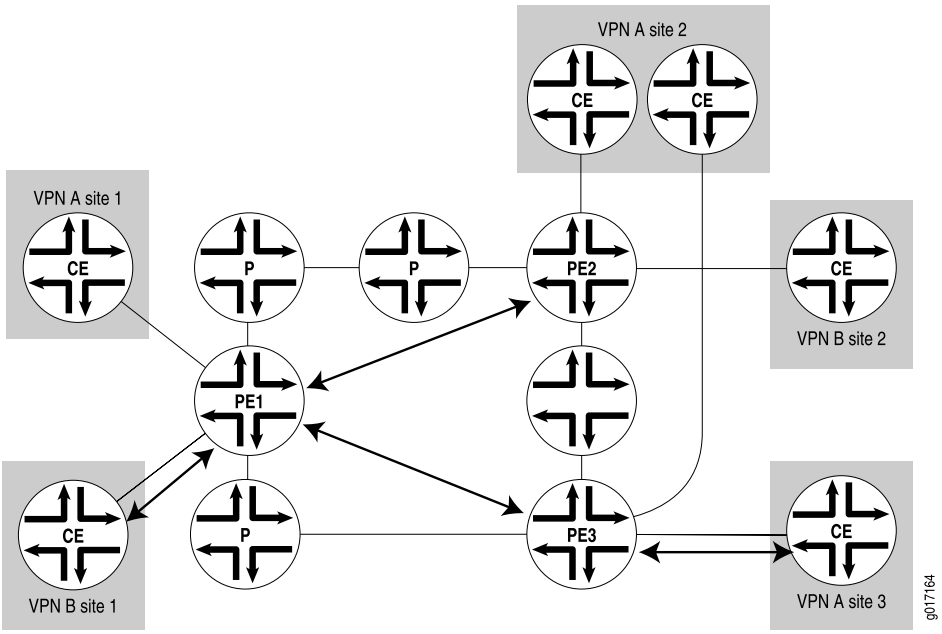
- Import policy—Applied to VPN-IPv4 routes learned from another PE router to determine whether the route should be added to the PE router's **bgp.l3vpn.0** routing table. Each routing instance on a PE router has a VRF import policy.
- Export policy—Applied to VPN-IPv4 routes that are announced to other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

Route Distribution Within a Layer 3 VPN

Within a VPN, the distribution of VPN-IPv4 routes occurs between the PE and CE routers and between the PE routers (see Figure 14 on page 156).

Figure 14: Route Distribution Within a VPN



This section discusses the following topics:

- Distribution of Routes from CE to PE Routers on page 156
- Distribution of Routes Between PE Routers on page 157
- Distribution of Routes from PE to CE Routers on page 158

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).

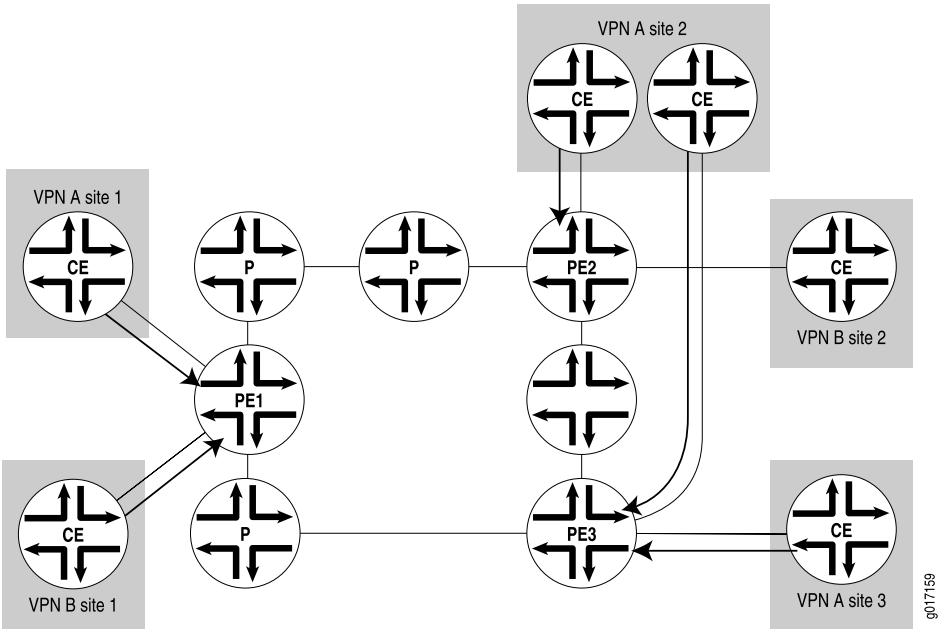
CE routers can communicate with PE routers using one of the following:

- OSPF
- RIP

- BGP
- Static route

Figure 15 on page 157 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 15: Distribution of Routes from CE Routers to PE Routers



Distribution of Routes Between PE Routers

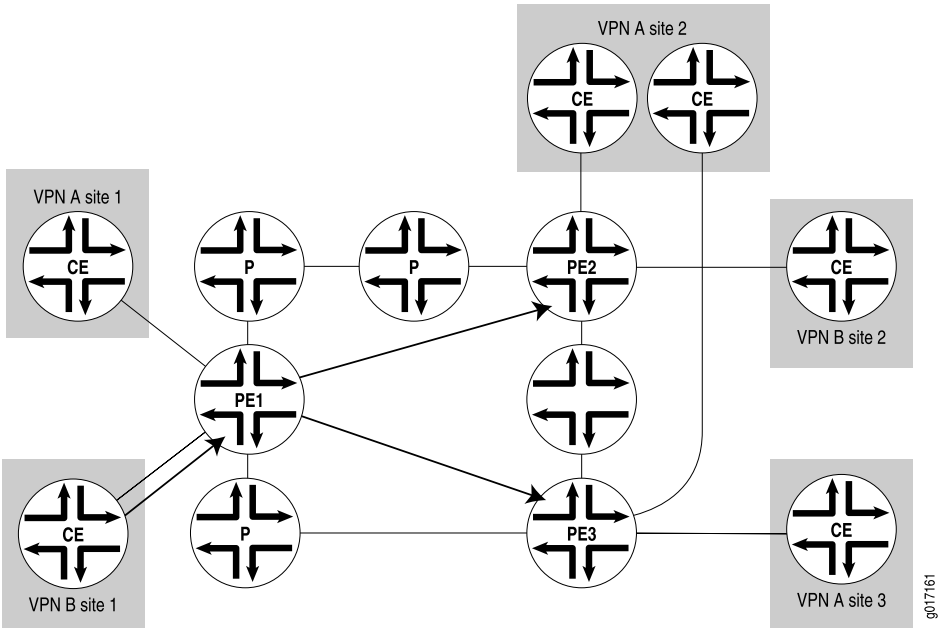
When one PE router receives routes advertised from a directly connected CE router, it checks the received route against the VRF export policy for that VPN. If it matches, the route is converted to VPN-IPv4 format—that is, the route distinguisher (route target) is added to the route. The PE router then announces 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. 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 16 on page 158 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 16: Distribution of Routes Between PE Routers



Distribution of Routes from PE to CE Routers

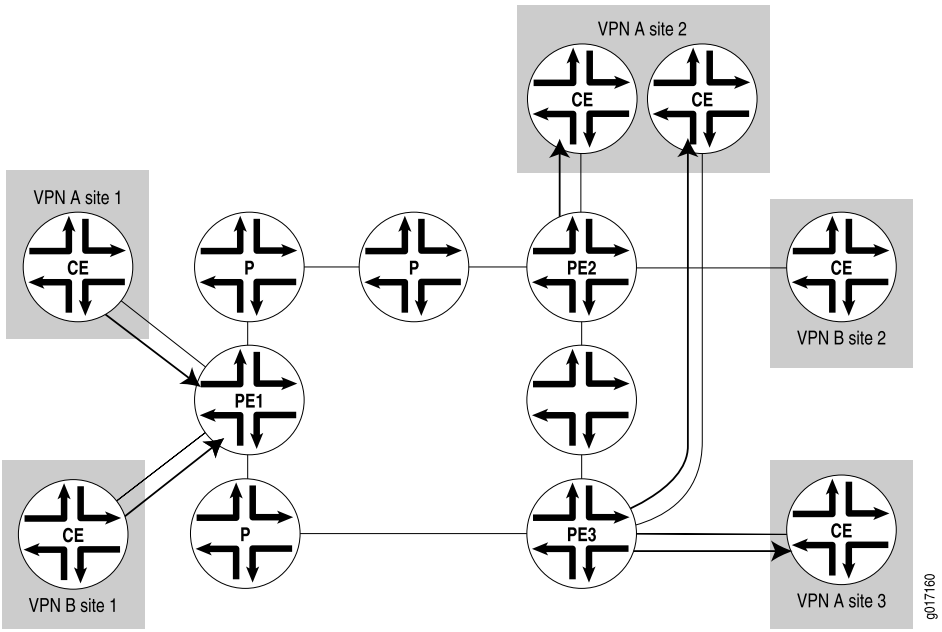
The remote PE router announces the routes in its VRF tables, which are in IPv4 format, to its directly connected CE routers.

PE routers can communicate with CE routers using one of the following routing protocols:

- OSPF
- RIP
- BGP
- Static route

Figure 17 on page 159 illustrates how the three PE routers announce their routes to their connected CE routers.

Figure 17: Distribution of Routes from PE Routers to CE Routers



Forwarding Across the Provider's Core Network

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 18 on page 160):

- 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 18: Using MPLS LSPs to Tunnel Between PE Routers

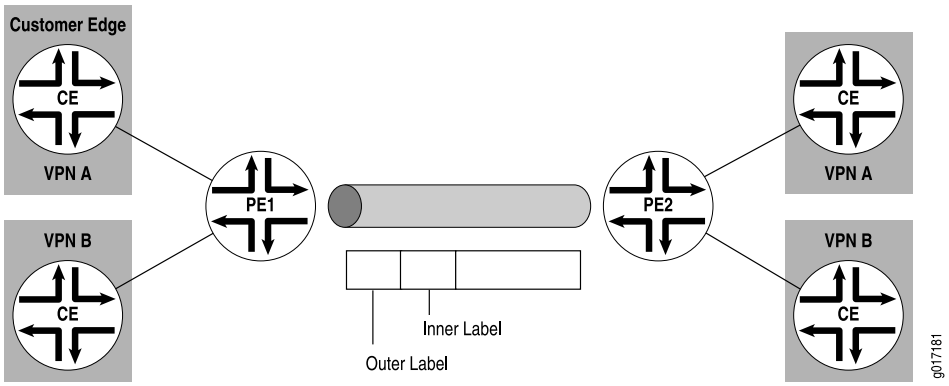
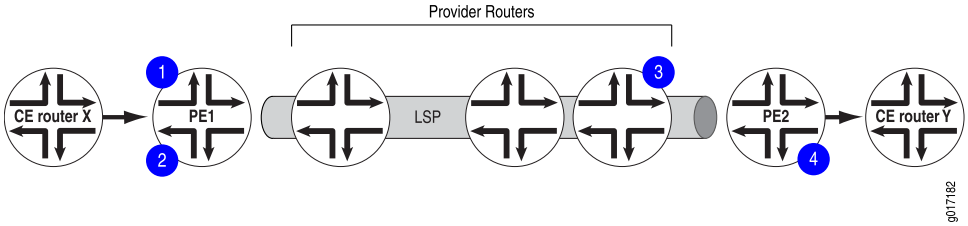


Figure 19 on page 160 illustrates how the labels are assigned and removed:

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 19: Label Stack



Routing Instances for VPNs

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.

Multicast over Layer 3 VPNs

You can configure multicast routing over a network running a Layer 3 VPN that complies with RFC 4364. This section describes this type of network application and includes these topics:

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 **vpn-group-address** statement at this hierarchy level, 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 master 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 master PIM instance. The set of master 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:

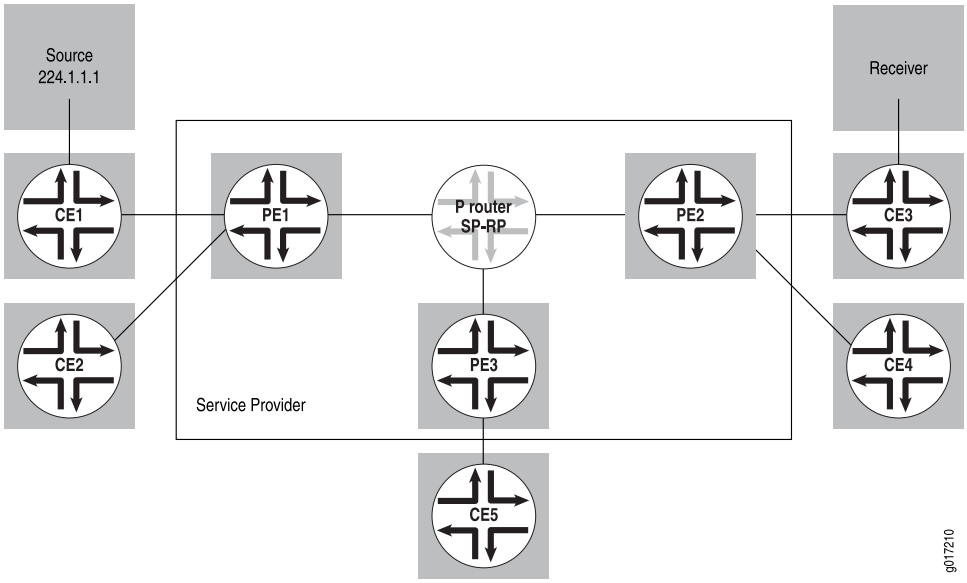
- P routers acting as RPs
- PE routers configured to run multicast routing
- CE routers acting as designated routers or as VPN-RPs

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*
- *Junos OS Multicast Protocols Configuration Guide*

The sections that follow describe the operation of a multicast VPN. Figure 20 on page 162 illustrates the network topology used.

Figure 20: 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 master 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 master PIM instance on each PE router handles the GRE encapsulated packet. Because the VPN group address is contained in the packet, the master 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 20 on page 162.

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.

Layer 3 VPN Standards

Layer 3 VPNs are defined in the following RFCs and IETF Internet drafts:

- RFC 1918, *Address Allocation for Private Internets*
- RFC 2685, *Virtual Private Networks Identifier*
- RFC 2858, *Multiprotocol Extensions for BGP4*
- RFC 4364, *BGP/MPLS IP Virtual Private Networks (VPNs)*
- RFC 4379, *Detecting Multi-Protocol Label Switched (MPLS) Data Plane Failures*
- 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)*

To access Internet RFCs and drafts, go to the IETF website at **http://www.ietf.org**.

CHAPTER 10

Configuring Layer 3 VPNs

This chapter describes how to configure Layer 3 VPNs, discussing the following topics:

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>;
    }
    pim {
        vpn-group-address address;
    }
    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*****]**

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

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.

Related Documentation

- Centralized Internet Access on page 351
- Configuring Hub-and-Spoke VPN Topologies: One Interface on page 240
- Configuring Hub-and-Spoke VPN Topologies: Two Interfaces on page 252
- Configuring Overlapping VPNs Using Automatic Route Export on page 302
- Configuring Overlapping VPNs Using Routing Table Groups on page 291
- Configuring a Full-Mesh VPN Topology with Route Reflectors on page 239
- Configuring a GRE Tunnel Interface Between PE Routers on page 306
- Configuring a GRE Tunnel Interface Between a PE and CE Router on page 312
- Configuring a Simple Full-Mesh VPN Topology on page 225
- Configuring an Application-Based Layer 3 VPN Topology on page 281
- Configuring an ES Tunnel Interface Between a PE and CE Router on page 315
- Configuring an LDP-over-RSVP VPN Topology on page 267
- Configuring an OSPF Domain ID for a Layer 3 VPN on page 285
- Distributed Internet Access on page 328
- Routing Internet Traffic Through a Separate NAT Device on page 344
- Routing VPN and Internet Traffic Through Different Interfaces on page 329
- Routing VPN and Internet Traffic Through the Same Interface Bidirectionally (VPN Has Private Addresses) on page 340
- Routing VPN and Internet Traffic Through the Same Interface Bidirectionally (VPN Has Public Addresses) on page 336
- Routing VPN and Outgoing Internet Traffic Through the Same Interface and Routing Return Internet Traffic Through a Different Interface on page 335
- Setting the Forwarding Class of the Ping Packets on page 40

Configuring Routing Between PE and CE Routers in Layer 3 VPNs

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 can configure only one type of routing.

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]**

  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]**

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]**

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:

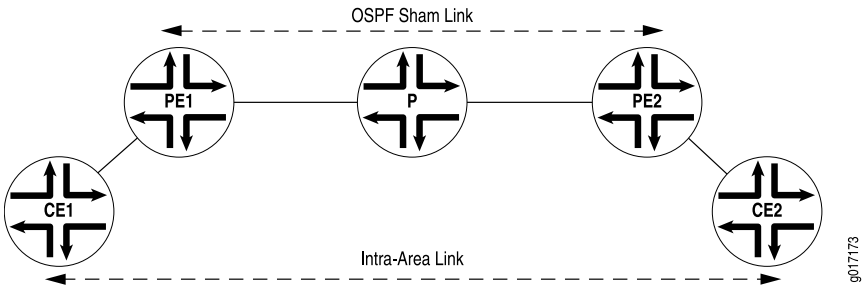
OSPF Sham Links Overview

Figure 21 on page 171 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 21 on page 171. 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 21: 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 1-.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 3 on page 173, a route with a null domain ID is handled differently from a route without any domain ID at all.

Table 3: 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

Table 3: 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	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
Type 3 route	Null	A.B.C.D	Type 5 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:

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 285.

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. These settings 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]**

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 an export policy, see the *Junos OS Policy Framework Configuration Guide*.

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*]**

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]**
- **[edit routing-instances *routing-instance-name* protocols]**

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

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]**

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 the *Junos OS Routing Protocols Configuration Guide*.

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*]**

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]**

For more information about configuring routing protocols and static routes, see the *Junos OS Routing Protocols Configuration Guide*.

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.

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.

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.
- **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.

Configuring Layer 3 VPNs to Carry IPv6 Traffic

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, 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 IPv6, see the *Junos OS Routing Protocols Configuration Guide*.

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]**

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]**

For complete configuration guidelines for this statement, see the *Junos OS Routing Protocols Configuration Guide*.

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***]**

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***]**

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***]**

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 Layer 3 VPNs to Carry IBGP Traffic

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.

Filtering Packets in Layer 3 VPNs Based on IP Headers

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;

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.

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

- Egress Filtering Options on page 184
- Support on Aggregated and VLAN Interfaces for IP-Based Filtering on page 185
- Support on ATM and Frame Relay Interfaces for IP-Based Filtering on page 185
- Support on Ethernet, SONET/SDH, and T1/T3/E3 Interfaces for IP-Based Filtering on page 186
- Support on SONET/SDH and DS3/E3 Channelized Enhanced Intelligent Queuing Interfaces for IP-Based Filtering on page 186
- Support on Multilink PPP and Multilink Frame Relay Interfaces for IP-Based Filtering on page 188
- Support for IP-Based Filtering of Packets with Null Top Labels on page 188
- General Limitations on IP-Based Filtering on page 189

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 4 on page 185.

Table 4: Support for Aggregated and VLAN Interfaces

Interfaces	J Series Router in Switching Mode	M Series Router Without an Enhanced FPC	M Series Router with an Enhanced FPC	M320 Router	T Series Router
Aggregated	N/A	No	Yes	Yes	Yes
VLAN	Yes	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 5 on page 185.

Table 5: Support for ATM and Frame Relay Interfaces

Interfaces	J Series Router	M Series Router Without an Enhanced FPC	M Series Router with an Enhanced FPC	M320 Router	T Series Router
ATM1	N/A	No	No	No	No
ATM2 intelligent queuing (IQ)	N/A	No	Yes	Yes	Yes
Frame Relay	Yes	No	Yes	Yes	Yes
Channelized	N/A	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 6 on page 186.

Table 6: Support for Ethernet, SONET/SDH, and T1/T3/E3 Interfaces

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

NOTE: The vrf-tabel-label statement is not supported on 4 port E3 IQ PICs.

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 available on M320 routers with Enhanced III FPCs as summarized in Table 7 on page 186.

Table 7: Support for Channelized IQE Interfaces on M320 Routers with Enhanced III FPCs

Interfaces	M320 Routers with Enhanced III FPCs
OC12	Yes
STM4	Yes

Table 7: Support for Channelized IQE Interfaces on M320 Routers with Enhanced III FPCs *(continued)*

Interfaces	M320 Routers with Enhanced III FPCs
OC3	Yes
STM1	Yes
DS3	Yes
E3	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 8 on page 188.

Table 8: Support for Multilink PPP and Multilink Frame Relay Interfaces

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

- 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 statement in a routing instance configuration that also includes a virtual loopback tunnel interface; the commit operation fails in this case.
- You cannot include the statement in source class usage (SCU) or destination class usage (DCU) configurations. For information about SCU and DCU configuration, see the *Junos OS Network Interfaces Configuration Guide*.
- You can include the statement in the configuration for Multilink Frame Relay (MLFR FRF.16) encapsulated PE-router-to-P-router interfaces only on J Series routers.
- 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
- 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:
  - 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

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 183) 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 *Junos OS Class of Service Configuration Guide*. 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 *Junos OS Class of Service Configuration Guide*.

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.

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-composite-nexthop** statement 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.

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]**

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]**

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**.

Configuring a VPN Tunnel for VRF Table Lookup

You can configure a VPN tunnel to facilitate VRF table lookup based on MPLS labels. You might want to enable this functionality to 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), or to perform egress filtering at the egress PE router.

For more information about VPN tunnels and VT interfaces, see the *Junos OS Services Interfaces Configuration Guide*.

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 “Pinging the Remote CE Router from the Local PE Router” on page 222.
- 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 VPNs” on page 40.

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 even configure a VRF loopback interface, the VRF routing instance uses Filter A.

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***]**

For more information about how to configure firewall filters, see the *Junos OS Policy Framework Configuration Guide*.

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 385.

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 *Junos OS Multicast Protocols Configuration 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*;

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. For more information about how to configure PIM and multicast in Junos, including an example of how to configure multicast over Layer 3 VPNs, see the *Junos OS Multicast Protocols Configuration Guide*.

Configuring 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.

For more information about the statements configured at the **[edit forwarding-options]** hierarchy level, see the *Junos OS Policy Framework Configuration Guide*.

Configuring GRE Tunnels for Layer 3 VPNs

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 22 on page 197) or to a remote CE router (as shown in Figure 23 on page 197).

Figure 22: GRE Tunnel Configured Between the Local CE Router and the PE Router

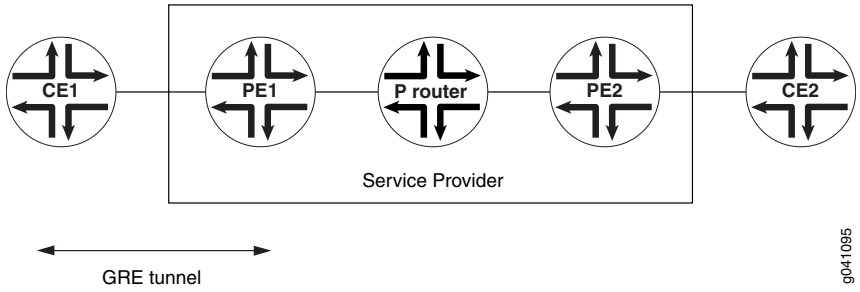
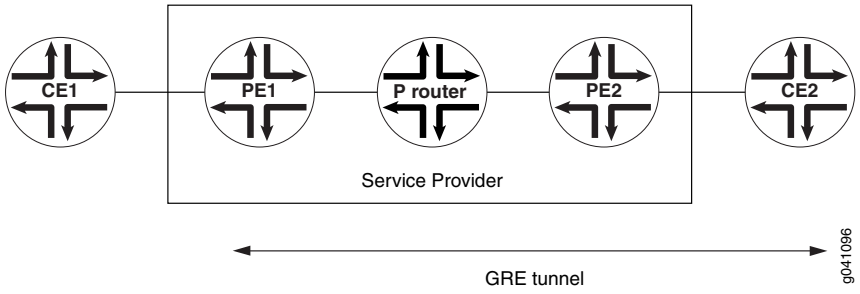


Figure 23: GRE Tunnel Configured Between the Remote CE Router and the PE Router



For more information about how to configure tunnel interfaces, see the *Junos OS Services Interfaces Configuration Guide*.

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 on page 198
- Configuring GRE Tunnels Dynamically on page 199

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 on page 198
- Configuring the GRE Tunnel Interface on the CE Router on page 199

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;
    tunnel-type gre;
}
```

You can include this statement at the following hierarchy levels:

- **[edit routing-options]**
- **[edit logical-systems *logical-system-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 dynamic-tunnels *tunnel-name*]**
- **[edit logical-systems *logical-system-name* routing-options dynamic-tunnels *tunnel-name*]**

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 dynamic-tunnels *tunnel-name*]**
- **[edit logical-systems *logical-system-name* routing-options dynamic-tunnels *tunnel-name*]**

Specify the type of tunnel to be dynamically created by including the **tunnel-type** statement. The only currently valid value is **gre** (for GRE tunnels).

```
tunnel-type gre;
```

You can include this statement at the following hierarchy levels:

- **[edit routing-options dynamic-tunnels *tunnel-name*]**
- **[edit logical-systems *logical-system-name* routing-options dynamic-tunnels *tunnel-name*]**

Configuring an ES Tunnel Interface for Layer 3 VPNs

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 on page 201
- Configuring the ES Tunnel Interface on the CE Router on page 202

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***]**

For more information about how to configure tunnel interfaces, see the *Junos OS Services Interfaces Configuration Guide*.

For more information about how to configure IPsec interfaces, see the *Junos OS System Basics Configuration Guide*.

Configuring IPsec Tunnels Instead of MPLS LSPs Between PE Routers in Layer 3 VPNs

A conventional Layer 3 BGP/MPLS VPN requires the configuration of MPLS label-switched paths (LSPs) between the PE routers. When a PE router receives a packet from a CE router, it performs a lookup in a specific VRF table for the IP destination address and obtains a corresponding MPLS label stack. The label stack is used to forward the packet

to the egress PE router, where the bottom label is removed and the packet is forwarded to the specified CE router.

You can provide Layer 3 BGP/MPLS VPN service without an MPLS backbone. Instead of configuring MPLS LSPs between the PE routers, you configure GRE and IPsec tunnels between the PE routers. The MPLS information for the VPN (the VPN label) is encapsulated within an IP header and an IPsec header. The source address of the IP header is the address of the ingress PE router. The destination address has the BGP next hop, the address of the egress PE router.

NOTE: The IPsec tunnel requires the use of an ES PIC. The GRE tunnel requires the use of a Tunnel Services PIC.

To configure IPsec between PE routers, follow these steps:

1. Configure an IPsec tunnel between the PE routers. The source address is that of the ingress PE router, and the destination address is that of the egress PE router:

```
es-interface-name {
  unit unit-number {
    tunnel {
      source source-address;
      destination destination-address;
    }
    family inet {
      ipsec-sa sa-esp-dynamic;
      address address;
    }
    family mpls;
  }
}
```

   You can include these statements at the following hierarchy levels:

   - **[edit interfaces]**
   - **[edit logical-systems *logical-system-name* interfaces]**

2. Configure IPsec on the PE router. For information about how to configure IPsec, see the *Junos OS System Basics Configuration Guide*.

3. Configure a GRE tunnel between the PE routers. Again, the source address is that of the ingress PE router, and the destination address is that of the egress PE router:

```
gr-interface-name {
  unit unit-number {
    family inet {
      address address;
    }
    family mpls;
    tunnel {
      source source-address;
      destination destination-address;
    }
```

```
    }
  }
```

You can include these statements at the following hierarchy levels:

- **[edit interfaces]**
- **[edit logical-systems *logical-system-name* interfaces]**

4. Configure BGP between the PE routers:

```
bgp {
  group pe {
    type internal;
    local-address local-address;
    family inet {
      unicast;
    }
    family inet-vpn {
      unicast;
    }
    peer-as as-number;
    neighbor address;
  }
}
```

You can include these statements at the following hierarchy levels:

- **[edit protocols]**
- **[edit logical-systems *logical-system-name* protocols]**

5. Configure the routing instance:

```
instance-type vrf;
interface interface-name;
route-distinguisher address;
vrf-import import-policy-name;
vrf-export export-policy-name;
protocols {
  bgp {
    group routing-instance-name {
      type external;
      peer-as as-number;
      as-override;
      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*]**

6. Configure the policy options:

```
policy-statement import-policy-name {
```



```
    term 1 {
      from {
        protocol bgp;
        community community-name;
      }
      then accept;
    }
    term 2 {
      then reject;
    }
  }
  policy-statement export-policy-name {
    term 1 {
      from protocol [ bgp direct ];
      then {
        community add community-name;
        accept;
      }
    }
    term 2 {
      then reject;
    }
  }
  community community-name members target:target;
```

You can include these statements at the following hierarchy levels:

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

7. Configure routing table groups to enable VPN route resolution in the **inet.3** routing table:

```
interface-routes {
  rib-group inet if-rib;
}
rib inet.3 {
  static {
    route BGP-address-for-remote-PE next-hop gre-interface-name;
  }
}
rib-groups {
  if-rib {
    import-rib [ inet.0 inet.3 ];
  }
}
```

You can include these statements at the following hierarchy levels:

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

Configuring Protocol-Independent Load Balancing in Layer 3 VPNs

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 on page 206
- Configuring Load Balancing and Routing Policies on page 207

Configuring Load Balancing for Layer 3 VPNs

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]**

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*]**

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.

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 191.

NOTE: You can include the **vpn-unequal-cost equal-external-internal** statement and the **l3vpn-composite-nexthop** statement 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.

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 *Junos OS Policy Framework Configuration Guide*.

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.

Related Documentation

- **as-path-compare on page 363**

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 *Junos OS Policy Framework Configuration 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]**

Accepting BGP Updates with Unique Inner VPN Labels in Layer 3 VPNs

For Layer 3 VPNs configured on Juniper Networks routers, the Junos software normally allocates one inner VPN label for each VPN network on the CE facing interfaces of a PE router. However, other vendors allocate one VPN label for each BGP route on the CE facing interfaces of a PE router. To account for this difference, configure the **l3vpn-composite-nexthop** statement at the **[edit routing-options]** hierarchy level on the Juniper Networks routers participating in a mixed vendor network. The **l3vpn-composite-nexthop** statement is disabled by default.

When you configure the **l3vpn-composite-nexthop** statement, the number of BGP routes with unique inner VPN labels that can be processed by a Juniper Networks router is increased substantially. This statement functions by combining the common elements of the BGP route updates associated with Layer 3 VPNs (hence the statement name). This functionality reduces the number of route updates and reduces the amount of state the router needs to maintain, leading to enhanced scaling and convergence performance on Juniper Networks routers participating in mixed vendor networks.

Juniper Networks recommends as a best practice that you configure the **l3vpn-composite-nexthop** statement whenever you have deployed Juniper Networks routers in mixed vendor networks to support Layer 3 VPNs.

You can configure the **l3vpn-composite-nexthop** statement on the following platforms:

- MX Series
- M120
- M320 with an Enhanced III FPC
- T Series

To accept larger numbers of Layer 3 VPN BGP updates with unique inner VPN labels, configure the **l3vpn-composite-nexthop** statement:

```
l3vpn-composite-nexthop;
```

You can include this statement at the following hierarchy levels:

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

NOTE: You can include the **vpn-unequal-cost equal-external-internal** statement and the **l3vpn-composite-nexthop** statement 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.

When you have configured the **l3vpn-composite nexthop** statement, you can determine whether or not a Layer 3 VPN route is a part of a composite next hop by examining the display output of the following commands:

- show route *route* extensive
- show route forwarding-table destination *destination* extensive

To enhance memory allocation to support a larger number of Layer 3 VPN labels, include the **vpn-label** statement at the **[edit chassis memory-enhanced]** hierarchy level. For more information on configuring more memory for Layer VPN labels, see the *Junos OS System Basics Configuration Guide*.

CHAPTER 11

Troubleshooting Layer 3 VPNs

This chapter discusses the following strategies and tools for troubleshooting Layer 3 virtual private network (VPN) configurations:

Diagnosing Common Problems

Problem 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
```

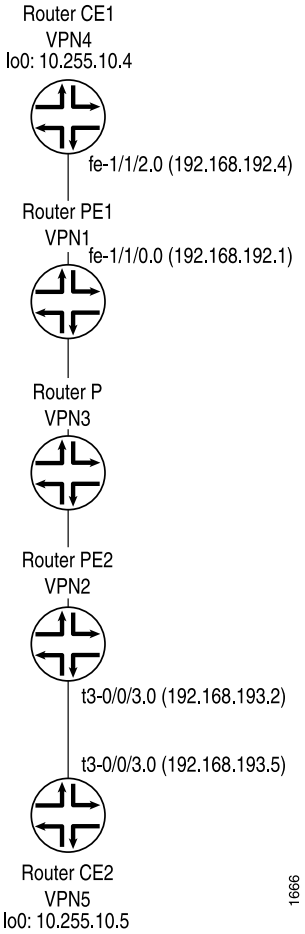
If the routes are not present, check the routing protocol configuration between the remote PE and CE routers, and make sure that peers and adjacencies (or static routes) between the PE and CE routers are correct.

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.

Troubleshooting Layer 3 VPNs Using ping and traceroute

This section provides examples of 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. The topology shown in Figure 24 on page 215 illustrates these commands.

Figure 24: Layer 3 VPN Topology for ping and traceroute Examples



Pinging the CE Router from Another CE Router

The following sections describe 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 24 on page 215 shows the topology referenced in the following examples:

- Pinging Router CE2 from Router CE1 on page 216
- Using traceroute from Loopback to Loopback on page 216
- Pinging Router CE1 from Router CE2 on page 216
- Using traceroute from Router CE2 to Router CE1 on page 216

Pinging Router CE2 from Router CE1

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
```

Using traceroute from Loopback to Loopback

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
```

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 24 on page 215 is not displayed.

Pinging Router CE1 from Router CE2

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
```

Using traceroute from Router CE2 to Router CE1

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

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 24 on page 215 shows the topology referenced in the following examples:

Pinging Router CE2 from Router CE1

Ping Router CE2 (VPN5) from Router CE1 (VPN4):

```
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
```

Using traceroute from Router CE1 to Router CE2

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
```

Pinging Router PE2 from Router CE1

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
```

Using traceroute from Router CE1 to Router PE2

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

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.

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;
          }
        }
      }
    }
```

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
```

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

From the loopback interfaces on the CE routers, you can ping the VPN interface on the directly connected PE router. Figure 24 on page 215 shows the topology referenced in the following examples:

- Pinging Router PE1 from the Loopback Interface on Router CE1 on page 220
- Using traceroute from the Loopback Interface on Router CE1 to PE1 on page 220
- Pinging Router PE2 from the Loopback Interface on Router CE2 on page 220
- Using traceroute from the Loopback Interface on Router CE2 to PE2 on page 220

Pinging Router PE1 from the Loopback Interface on Router CE1

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
```

Using traceroute from the Loopback Interface on Router CE1 to PE1

To determine the path from the loopback interface on Router CE1 to the VPN interfaces on Router PE1, use the **traceroute** command:

```
user@vpn4> traceroute 192.168.192.1 source 10.255.10.4
traceroute to 192.168.192.1 (192.168.192.1) from 10.255.10.4, 30 hops max, 40 byte
 packets
 1  vpn1-fe-110.isp-core.net (192.168.192.1)  0.828 ms  0.657 ms  1.972 ms
```

Pinging Router PE2 from the Loopback Interface on Router CE2

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
```

Using traceroute from the Loopback Interface on Router CE2 to PE2

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

From the VPN and loopback interfaces on the PE routers, you can ping the VPN interface on the directly connected CE router. Figure 24 on page 215 shows the topology referenced in the following examples:

- Pinging the VPN Interface on Router CE1 from Router PE1 on page 221
- Pinging the Loopback Interface on Router CE1 from Router PE1 on page 221

Pinging the VPN Interface on Router CE1 from Router PE1

From the VPN interface on the PE router, you can ping the VPN or loopback interface on the directly connected CE router.

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
```

Pinging the Loopback Interface on Router CE1 from Router PE1

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
```

Using traceroute from Router PE1 to Router CE1

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
```

Pinging the VPN Interface on Router CE2 from Router PE2

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
```

Pinging the Loopback Interface on Router CE2 from Router PE2

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
```

Using traceroute from Router PE2 to Router CE2

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

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 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*.

Limitation on Pinging a Remote CE Router from a PE Router

If you attempt to ping a remote CE router from a PE router, ICMP echo requests are sent from the PE router, with the PE router's VPN interface as the source. Other PE routers have a route back to that address with a VPN label. When the echo replies return, they include a label. The PE router pops the VPN label and sends the packet from the VPN interface to the local CE router. The local CE router sends it back to the PE router, its actual destination.

When a Juniper Networks routing platform receives a labeled packet, the label is popped (depending on the label operation specified), and the packet is forwarded to an interface, even if the packet is destined for that particular PE router. Labeled packets are not analyzed further for the IP information under the label.

If there is a problem with the connection to the local CE router, packets are sent out but do not return to the PE router, and the ping fails. If the connection between your PE router and local CE router is down, sending a ping to the remote CE router fails even though the connection to the remote CE router might be functional.

Troubleshooting Inconsistently Advertised Routes from Gigabit Ethernet Interfaces

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:

- 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.

- 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 "Support on Ethernet, SONET/SDH, and T1/T3/E3 Interfaces for IP-Based Filtering" on page 186 for information about support for the **vrf-table-label** statement over Ethernet and SONET/SDH interfaces.

CHAPTER 12

Layer 3 VPN Configuration Examples

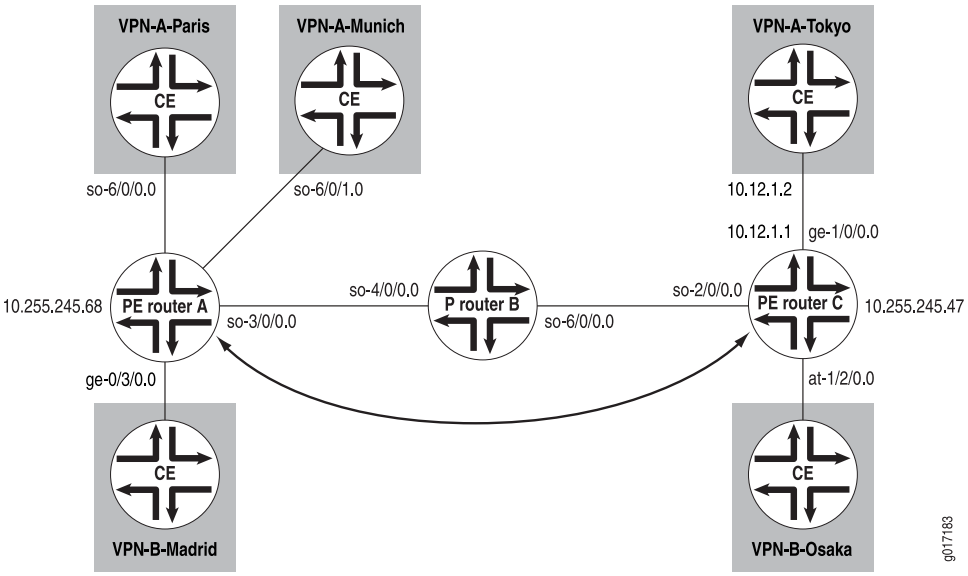
This chapter provides the following examples of Layer 3 virtual private network (VPN) configurations:

Configuring a Simple Full-Mesh VPN Topology

This example shows how to set up a simple full-mesh service provider VPN configuration, which consists of the following components (see Figure 25 on page 226):

- 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 25: 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 master routing table. (For routing platforms running Junos OS, the master 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 25 on page 226.

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 master 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-0/0/0.0;
            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]
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]
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;
    }
  }
}
```

Master 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)

Master 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;
        }
    }
```

```
}
```

Master 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 225) in which one of the PE routers is a BGP route reflector. In this variation, Router C in Figure 25 on page 226 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;
    }
```

```
    }
}
```

For the complete configuration example of Router C, see “Configuring a Full-Mesh VPN Topology with Route Reflectors” on page 239.

Configuring Hub-and-Spoke VPN Topologies: One Interface

Use a one-interface configuration to advertise a default route from a hub or hubs.

Figure 26: Example of a Hub-and-Spoke VPN Topology with One Interface

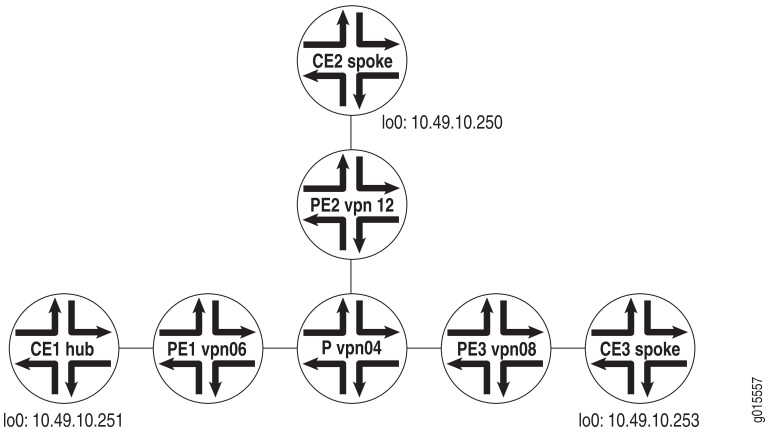


Figure 26 on page 240 illustrates a Layer 3 VPN hub-and-spoke application where there is only one interface between the hub CE (CE1) and the hub PE (PE1). This is the recommended way of configuring hub-and-spoke topologies.

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 252 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 26 on page 240:

- Configuring Hub CE1 on page 241
- Configuring Hub PE1 on page 242
- Configuring the P Router on page 242
- Configuring Spoke PE2 on page 243
- Configuring Spoke PE3 on page 244
- Configuring Spoke CE2 on page 246
- Configuring Spoke CE3 on page 246
- Enabling Egress Features on the Hub PE Router on page 248

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-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 {
      encapsulation frame-relay;
      unit 0 {
        dlci 16;
        family inet {
          address 10.49.4.1/30;
        }
      }
    }
    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;
        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;
    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 PE2 (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 248.

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 240. This example also uses the topology illustrated in Figure 26 on page 240.

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 240 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;
      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;
      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 the routing instances configuration guidelines in the *Junos OS Routing Protocols Configuration Guide* 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 the routing instances configuration guidelines in the *Junos OS Routing Protocols Configuration Guide*

for more information about using the **auto-export** statement. See Configuring Overlapping VPNs Using Automatic Route Export 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.

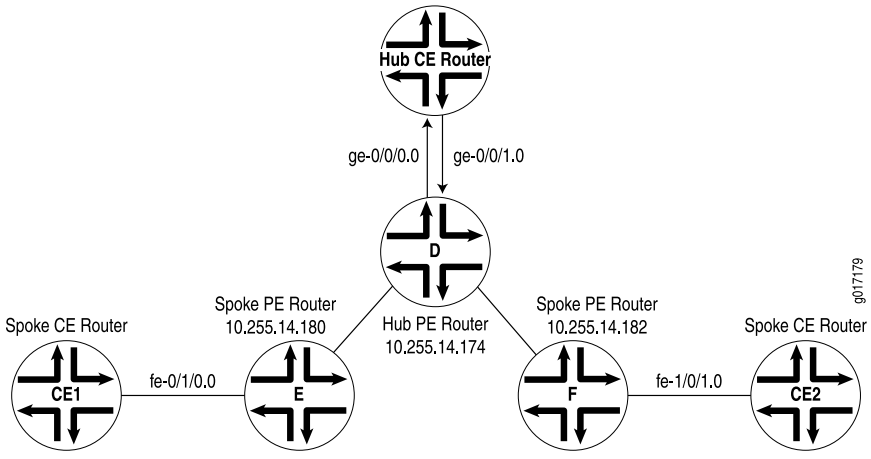
Configuring Hub-and-Spoke VPN Topologies: Two Interfaces

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 27 on page 252):

- 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 27: Example of a Hub-and-Spoke VPN Topology with Two Interfaces



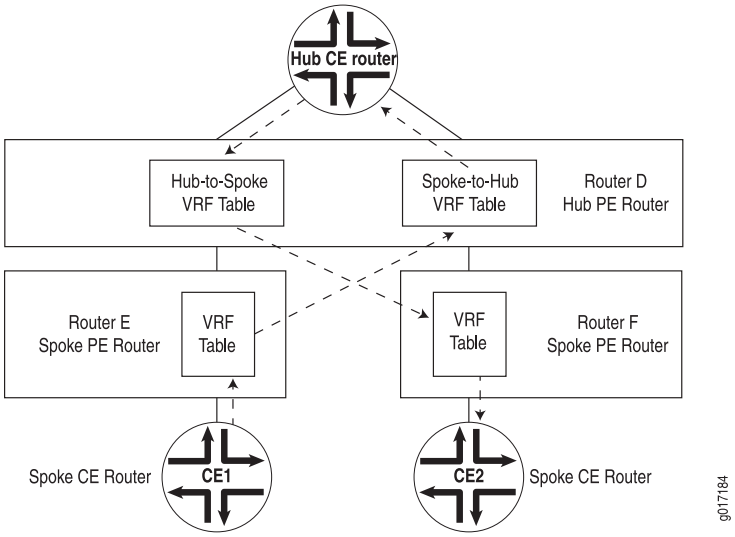
In this configuration, route distribution from spoke CE Router CE1 occurs as follows:

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 28 on page 254 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 262, consolidates the statements needed to configure VPN functionality for each of the service provider routers shown in Figure 27 on page 252.

Figure 28: 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 on page 254
- Configuring LDP on the Hub-and-Spoke PE Routers on page 255
- Configuring IBGP on the PE Routers on page 255
- Configuring VPN Routing Instances on the Hub-and-Spoke PE Routers on page 256
- Configuring VPN Policy on the PE Routers on page 259
- Hub-and-Spoke VPN Configuration Summarized by Router on page 262

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 master 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 {
            export redistribute-vpn;
            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 10.255.1.174:65535;
      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 10.255.14.80:65535;
    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 10.255.14.182:65535;
    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

```
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;
  }
}
```

Instance Routing Protocol

```
protocols {
  ospf {
    export redistribute-vpn;
    area 0.0.0.0 {
      interface ge-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 10.255.1.174:65535;
  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 (Master Instance)

```
routing-options {
  autonomous-system 1 loops 1;
}
```

Protocols (Master 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 10.255.14.80:65535;
    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 (Master Instance)

```
routing-options {
  autonomous-system 1 loops 1;
}
```

Protocols (Master 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 10.255.14.182:65535;
        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 (Master Instance)

```
routing-options {
    autonomous-system 1 loops 1;
}
```

Protocols (Master 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

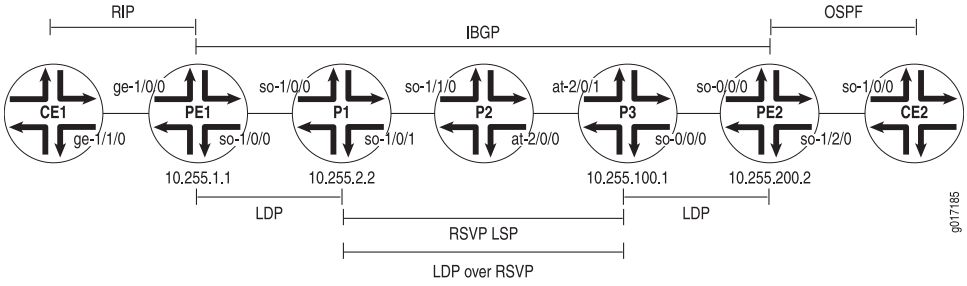
```
policy-options {
  policy-statement hub {
    term a {
      from {
        protocol bgp;
        community hub;
      }
      then accept;
    }
    term b {
      then reject;
    }
  }
  policy-statement spoke {
    term a {
      from protocol ospf;
      then {
        community add spoke;
        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;
}
```

Configuring an LDP-over-RSVP VPN Topology

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 29 on page 267):

- 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 29: 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**.

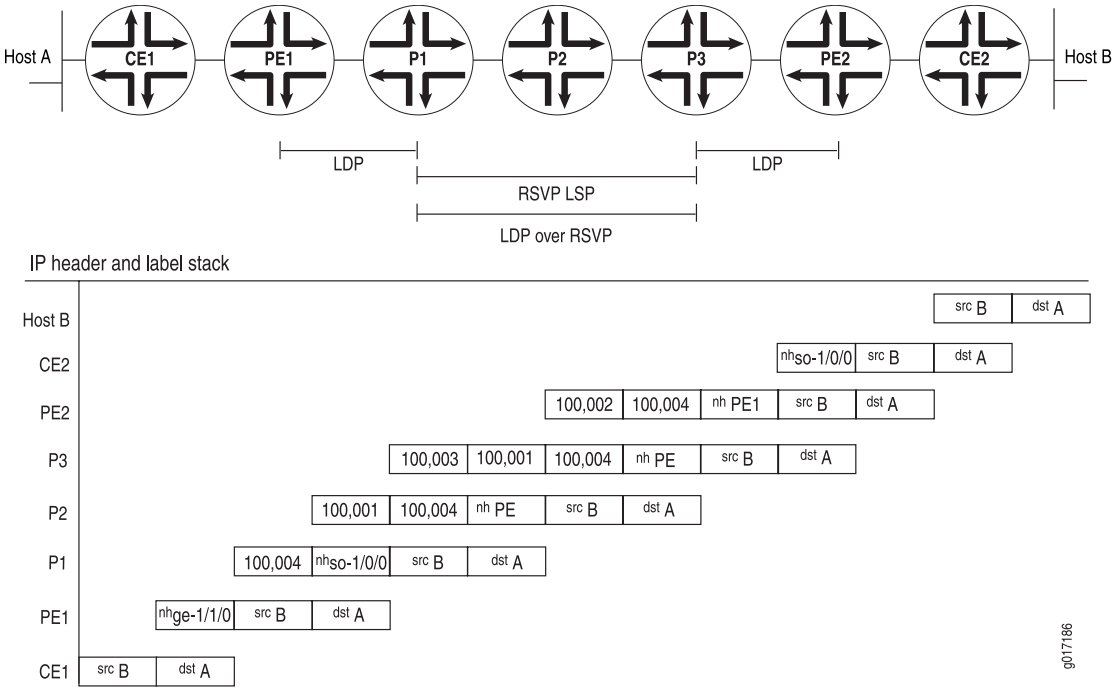
A similar set of operations occurs for packets sent from Router CE1 that are destined for Router CE2.

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 30 on page 269).

- 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 30: Label Pushing and Popping



For a packet sent from Host B in Figure 30 on page 269 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 29 on page 267.

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 master 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;
  }
}
```

Master 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

Master 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

Master 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

Master 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;
  }
}
```

Master 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

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:

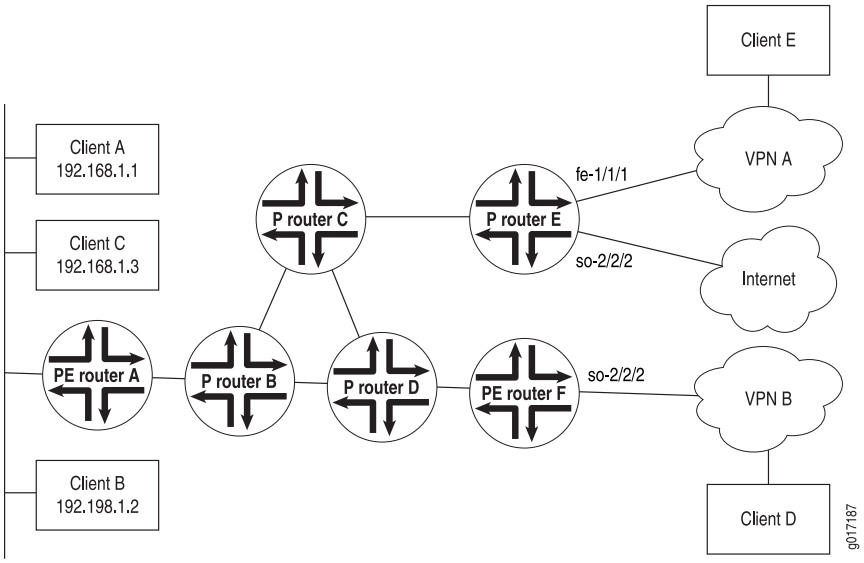
- Client A sends traffic to Client E over VPN A with a return path that also uses VPN A (using the VPN's VRF table).
- Client B sends traffic to Client D over VPN B with a return path that uses standard destination-based routing (using the **inet.0** routing table).
- Clients B and C send traffic to 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 31 on page 282 illustrates the network topology used in this example.

Figure 31: Application-Based Layer 3 VPN Example Configuration



- Configuration on Router A on page 282
- Configuration on Router E on page 284
- Configuration on Router F on page 285

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;
      }
    }
  }
}
```

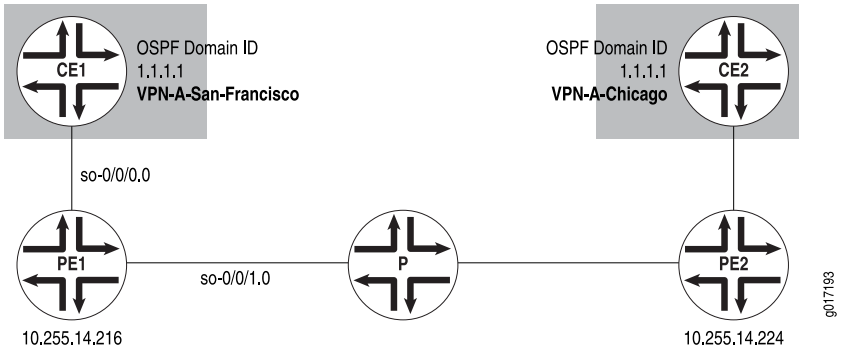
Configuring an OSPF Domain ID for a Layer 3 VPN

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.

For more information about OSPF domain IDs and Layer 3 VPNs, see “Configuring an OSPF Domain ID” on page 173.

Figure 32 on page 286 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 32: Example of a Configuration Using an OSPF Domain ID



For configuration information, see the following sections:

- Configuring Interfaces on Router PE1 on page 286
- Configuring Routing Options on Router PE1 on page 287
- Configuring Protocols on Router PE1 on page 287
- Configuring Policy Options on Router PE1 on page 287
- Configuring the Routing Instance on Router PE1 on page 288
- Configuration Summary for Router PE1 on page 289

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 69;
}
```

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:1.1.1.1:0];
   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 32 on page 286, 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;
      autonomous-system 69;
    }
    protocols {
      ospf {
        domain-id 1.1.1.1;
        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 69;
}
```

Configure Protocols

```
protocols {
  mpls {
    interface so-0/0/0.0;
  }
  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-B members [ target:10.255.14.216:11domain-id:1.1.1.1:0 ];
  community import-target-VPN-B 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;
      autonomous-system 69;
    }
    protocols {
      ospf {
        domain-id 1.1.1.1;
        export vpn-import-VPN-A;
```

```
            area 0.0.0.0 {
              interface so-0/0/0.0;
            }
          }
        }
      }
    }
```

Configuring Overlapping VPNs Using Routing Table Groups

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.

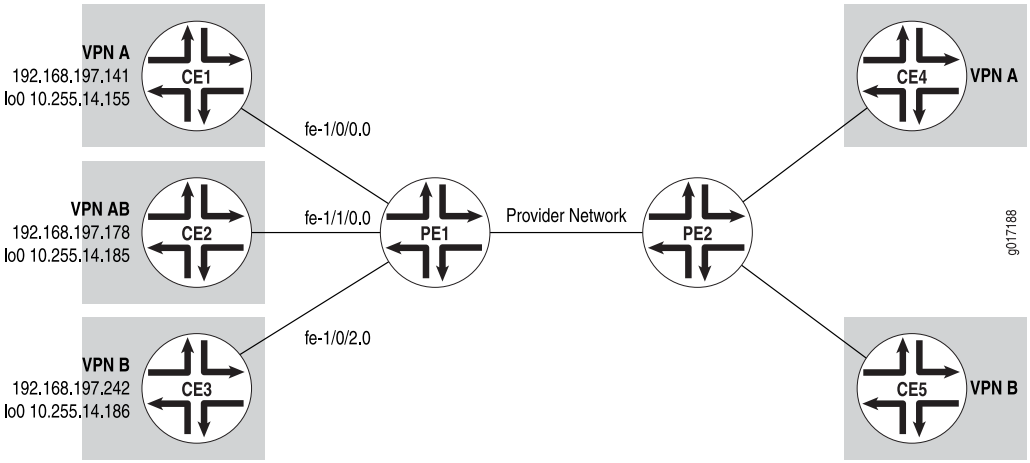
You define routing table groups at the **[edit routing-options]** hierarchy level for the default instance. You cannot configure routing table groups at the **[edit routing-instances routing-options]** hierarchy level; doing so results in a commit error.

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 33 on page 292 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 33: 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 on page 292
- Configuring Static Routes Between the PE and CE Routers on page 293
- Configuring BGP Between the PE and CE Routers on page 298
- Configuring OSPF Between the PE and CE Routers on page 299
- Configuring Static, BGP, and OSPF Routes Between PE and CE Routers on page 301

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 page 293
- Configuring the Routing Instance for VPN AB on page 294
- Configuring the Routing Instance for VPN B on page 294
- Configuring VPN Policy on page 295

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 293, 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 293 and “Configuring BGP Between the

PE and CE Routers" on page 298, 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

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 291.

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 on page 303
- Configuring Overlapping VPNs and Additional Tables on page 304
- Configuring Automatic Route Export for All VRF Instances on page 305

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:

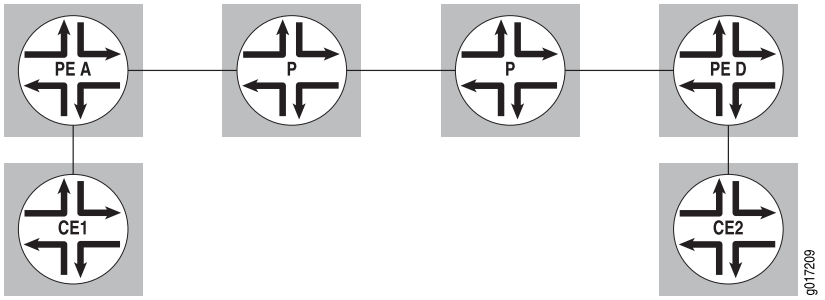
```
[edit]
groups {
    vrf-export-on {
        routing-instances {
            <*> {
                routing-options {
                    auto-export;
                }
            }
        }
    }
```

```
}
apply-groups vrf-export-on;
```

Configuring a GRE Tunnel Interface Between PE Routers

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 34 on page 306. The P routers shown in this illustration do not run MPLS.

Figure 34: 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 on page 306
- Configuring the Routing Instance on Router D on page 307
- Configuring MPLS, BGP, and OSPF on Router A on page 307
- Configuring MPLS, BGP, and OSPF on Router D on page 308
- Configuring the Tunnel Interface on Router A on page 308
- Configuring the Tunnel Interface on Router D on page 308
- Configuring the Routing Options on Router A on page 309
- Configuring the Routing Options on Router D on page 309
- Configuration Summary for Router A on page 309
- Configuration Summary for Router D on page 311

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;
  }
}
```

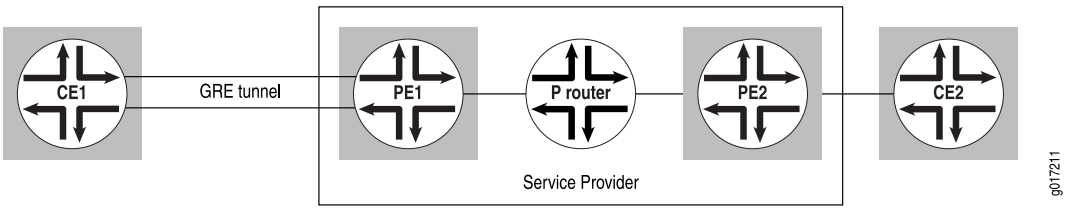
Configure the Routing Options

```
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

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 35 on page 312.

Figure 35: 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 on page 313
- Configuring the Routing Instance with the Encapsulating Interface on page 314
- Configuring the GRE Tunnel Interface on Router CE1 on page 315

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.

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 on page 314
- Configuring the GRE Tunnel Interface on Router PE1 on page 314
- Configuring the Encapsulation Interface on Router PE1 on page 315

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;
    }
  }
```

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

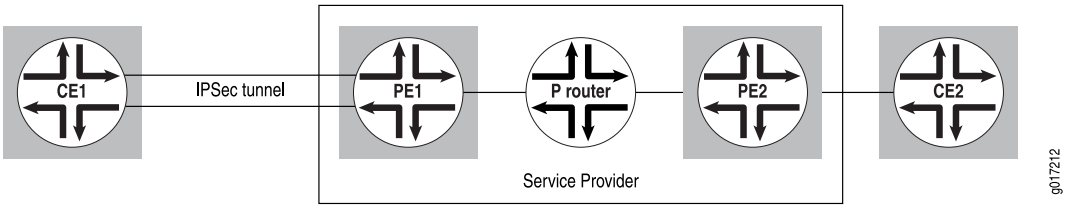
Configure the GRE tunnel interface on Router CE1:

```
[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;
  }
}
```

Configuring an ES Tunnel Interface Between a PE and CE Router

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 36 on page 315.

Figure 36: ES Tunnel Interface (IPsec Tunnel)



To configure this example, you perform the steps in the following sections:

- Configuring IPsec on Router PE1 on page 316
- Configuring the Routing Instance Without the Encapsulating Interface on page 316
- Configuring the Routing Instance with the Encapsulating Interface on page 317
- Configuring the ES Tunnel Interface on Router CE1 on page 318
- Configuring IPsec on Router CE1 on page 319

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 on page 316
- Configuring the ES Tunnel Interface on Router PE1 on page 317
- Configuring the Encapsulating Interface for the ES Tunnel on page 317

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 on page 317
- Configuring the ES Tunnel Interface on Router PE1 on page 318
- Configuring the Encapsulating Interface on Router PE1 on page 318

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";
            }
         }
      }
   }
}
```

Example: Disabling Normal TTL Decrementing in a VRF Routing Instance

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

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.

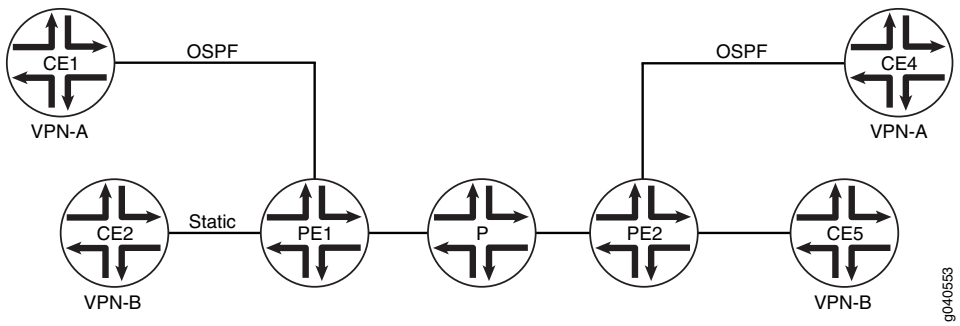
When you change the TTL propagation behavior, old next hops for VRF routes are deleted from the inet.3 routing table and new next hops are added.

You need only configure the **vrf-propagate-ttl** or **no-vrf-propagate-ttl** statement on the ingress routers.

Topology Diagram

Figure 37 on page 320 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 37: Disabling TTL Propagation for a Single VPN



Configuration

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 37 on page 320.

```
[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.

Related Documentation

- Disabling Normal TTL Decrementing in the *Junos OS MPLS Applications Configuration Guide*

CHAPTER 13

Layer 3 VPN Internet Access Examples

There are several ways to configure a PE router to provide CE routers access to the Internet. These types of access are described in the following sections:

Non-VRF Internet Access

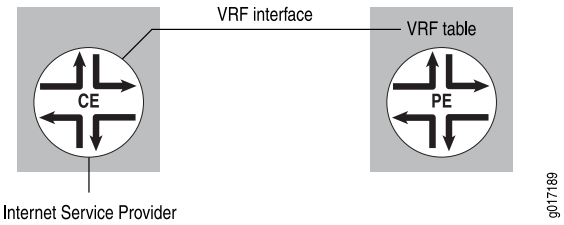
Junos OS supports Internet access from a Layer 3 virtual private network (VPN). This chapter provides examples that demonstrate how to configure a provider edge (PE) router to provide Internet access to customer edge (CE) routers in a VPN. The method you use depends on the needs and specifications of the individual network. To provide Internet access through a Layer 3 VPN, you need to configure policies on the PE router. 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 38 on page 328).

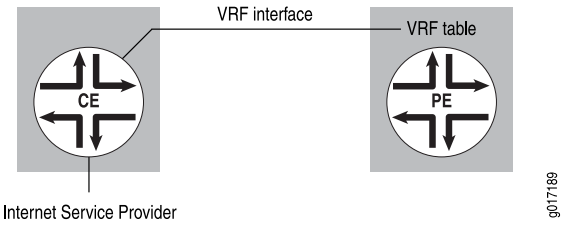
Figure 38: PE Router Does Not Provide Internet Access



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 39 on page 328).

Figure 39: PE Router Connects to a Router Connected to the Internet



Distributed Internet Access

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.

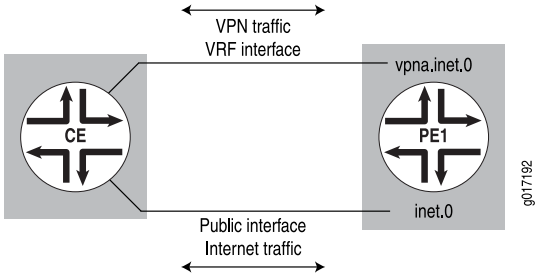
When accessing the Internet from a VPN, Network Address Translation (NAT) must be performed between the VPN's private addresses and the public addresses used on the Internet unless the VPN is using the public address space. This section includes several examples of how to provide Internet access for VPNs, most of which require that the CE routers perform the address translation. The "Routing Internet Traffic Through a Separate NAT Device" on page 344 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

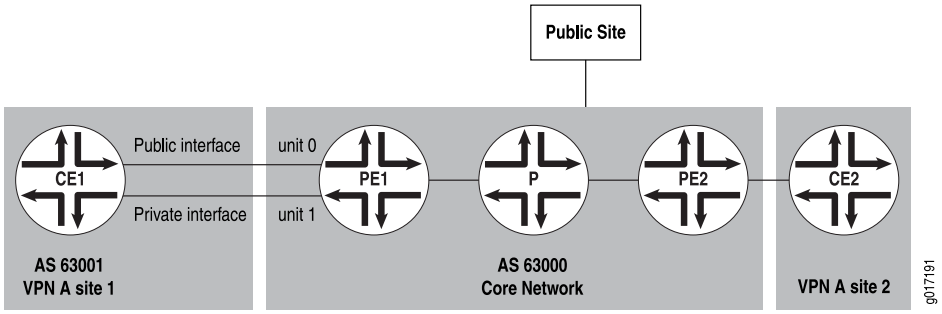
In this example, VPN and Internet traffic are routed through different interfaces. The CE router sends the VPN traffic through the VPN interface and sends the Internet traffic through a separate interface that is part of the main routing table on Router PE1 (the CE router can use either one physical interface with two logical units or two physical interfaces). NAT also occurs on the CE router (see Figure 40 on page 329).

Figure 40: 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 41 on page 329 illustrates the PE router's VPN configuration.

Figure 41: 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 the 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

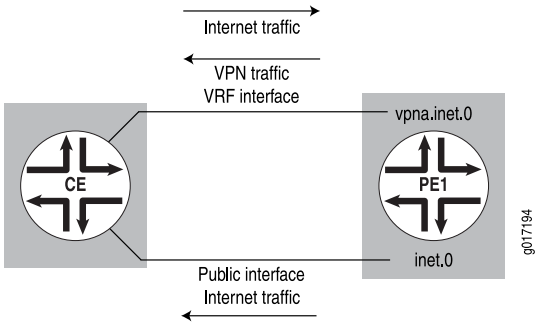
```
policy-statement vpna-import {
  term a {
    from {
      protocol bgp;
      community vpna-comm;
    }
    then accept;
  }
  term b {
    then reject;
  }
```

```
}
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 42 on page 335). The CE router still performs NAT functions.

Figure 42: 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 on page 335

Configuration for Router PE1

This example has the same configuration as Router PE1 in "Routing VPN and Internet Traffic Through Different Interfaces" on page 329. It uses the topology shown in Figure 41 on page 329. 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 {
```

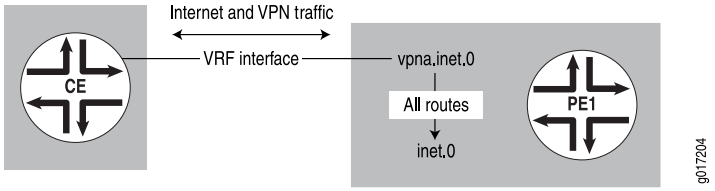
```
vpna {
  instance-type vrf;
  interface t3-0/2/0.0;
  route-distinguisher 10.255.14.171:100;
  vrf-import vpna-import;
  vrf-export vpna-export;
  routing-options {
    static {
      route 0.0.0.0/0 next-table inet.0;
    }
  }
  protocols {
    bgp {
      group to-CE1 {
        peer-as 63001;
        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" on page 329) 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)

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 43 on page 336).

Figure 43: 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" on page 329. 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 t3-0/2/0.0;
   }
   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" on page 329. 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 t3-0/2/0.0;
  }
  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;
```

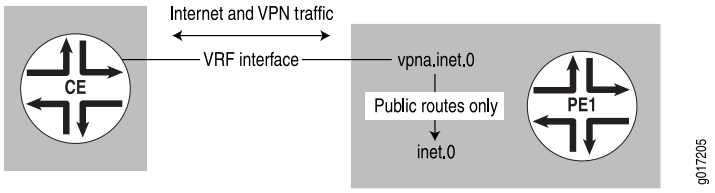
```
        }
      }
      protocols {
        bgp {
          group to-CE1 {
            family inet {
              unicast {
                rib-group vpna-to-inet0;
              }
            }
            peer-as 63001;
            neighbor 192.168.197.14;
          }
        }
      }
    }
  }
```

Routing VPN and Internet Traffic Through the Same Interface Bidirectionally (VPN Has Private Addresses)

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 336, 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 44 on page 340).

Figure 44: 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 page 341
- Configuring a Routing Instance for Router PE1 on page 341

- Configuring Policy Options for Router PE1 on page 342
- Traffic Routed by the Same Interface Bidirectionally (VPN Has Private Addresses): Configuration Summarized by Router on page 342

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-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

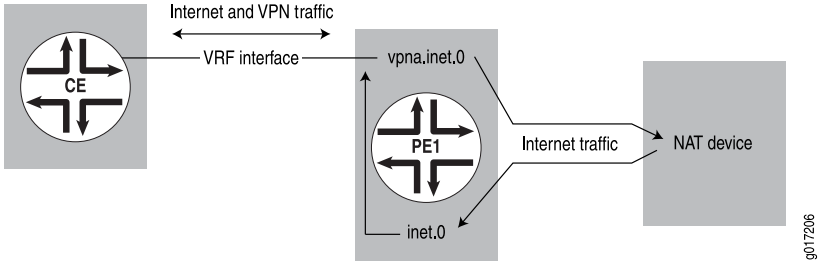
```
[edit]
policy-options {
  policy-statement import-public-addr-to-inet0 {
    term a {
      from {
        protocol bgp;
        rib vpna.inet.0;
        community [ public-comm private-comm ];
      }
      then accept;
    }
    term b {
      from {
        protocol bgp;
        community public-comm;
      }
      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

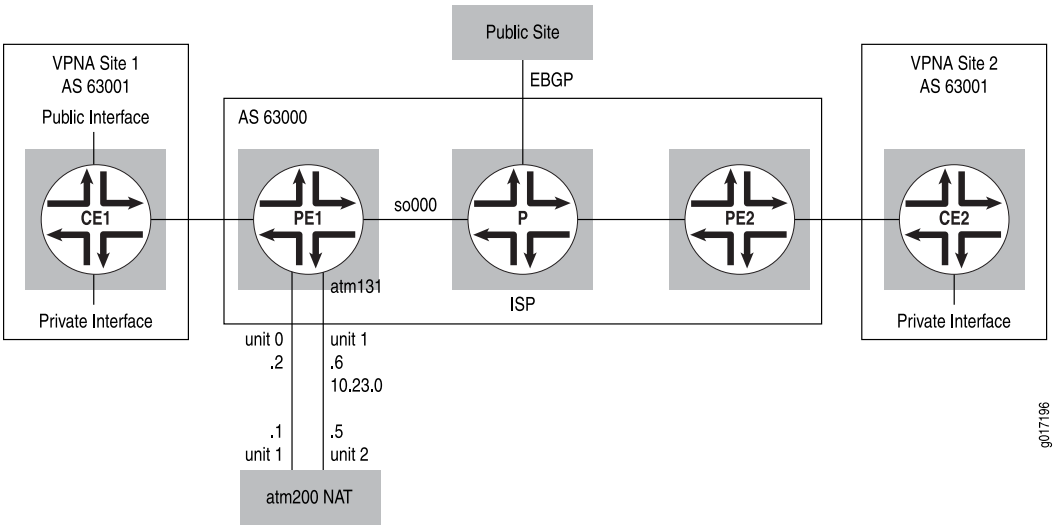
In this example, the CE router does not perform NAT. It sends both VPN and Internet traffic over the same interface to the PE router. The PE router is connected to an NAT device by means of two interfaces. One interface is configured in the PE router's VRF table and points to a VPN interface on the NAT 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 45 on page 344).

Figure 45: Internet Traffic Routed Through a Separate NAT Device



This example's topology expands upon that illustrated in Figure 41 on page 329. 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 46 on page 344).

Figure 46: Internet Traffic Routed Through a NAT Example Topology



The following sections show how to route Internet traffic through a separate NAT device:

Configuring Interfaces on Router PE1

Configure an interface for VPN traffic to and from Router CE1, an interface for VPN traffic to and from the NAT device, and an interface for Internet traffic to and from the NAT device:

```
[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;
            }
        }
    }
    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

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

Configure MPLS, BGP, IS-IS, and LDP on Router PE1. For the MPLS configuration, include the NAT device's VPN interface in the VRF table. As part of the BGP configuration, include a policy to advertise the public IP address pool:

```
[edit]
protocols {
   mpls {
      interface t3-0/2/0.0;
      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;
   }
}
```

Configuring a Routing Instance for Router PE1

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;
                }
            }
        }
    }
}
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;
}
```

Traffic Routed by Separate NAT Device: 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;
      }
    }
  }
  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;
        }
      }
    }
  }
```

```
}
```

Routing Options

```
routing-options {
  static {
    route 10.12.1.0/24 next-hop 10.23.0.5;
  }
}
```

Routing Protocols

```
protocols {
  mpls {
    interface t3-0/2/0.0;
    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;
  }
}
```

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 {
          peer-as 63001;
```

```
            neighbor 192.168.197.14;
          }
        }
      }
    }
```

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

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.

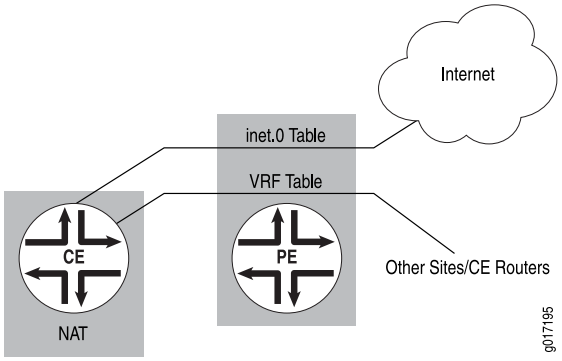
This section includes the following examples:

- Routing Internet Traffic Through a Hub CE Router on page 351
- Routing Internet Traffic Through Multiple CE Routers on page 355

Routing Internet Traffic Through a Hub CE Router

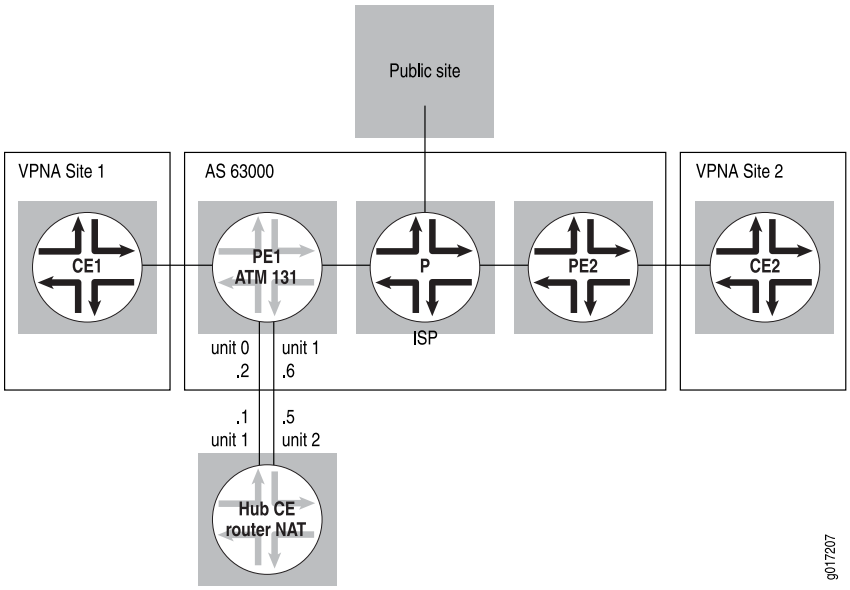
In this example, Internet traffic is routed through a hub CE router. The hub CE router has two interfaces to the hub PE router: a VPN interface and a public interface. It performs NAT on traffic forwarded from the hub PE router through the VPN interface and forwards that traffic from its public interface 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 47 on page 351).

Figure 47: 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 344. The difference is that Router PE1 is configured to announce a static default route to the other CE routers (see Figure 48 on page 352).

Figure 48: 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 on page 352
- Configuring Policy Options on Router PE1 on page 353
- Internet Traffic Routed by a Hub CE Router: Configuration Summarized by Router on page 354

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 344. 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

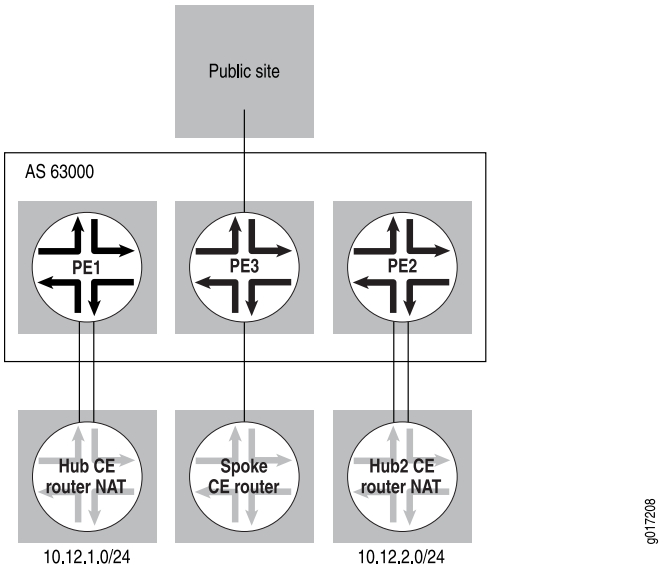
```
policy-options {
  policy-statement vpna-export {
    term a {
      from protocol bgp;
      then {
        community add vpna-comm;
        accept;
      }
    }
    term b {
      from {
        protocol static;
        route-filter 0.0.0.0/0 exact;
      }
      then {
        community add vpna-comm;
        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" on page 351. 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 49 on page 355).

Figure 49: 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 351, 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 356:

```
[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 351. 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** (see "Policy Options" on page 359).

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;
}
```

CHAPTER 14

Summary of Layer 3 VPN Configuration Statements

The following section explains the major **routing-instances** configuration statements that apply specifically to Layer 3 virtual private networks (VPNs).

as-path-compare

Syntax as-path-compare;

Hierarchy Level [edit logical-systems *logical-system-name* routing-instances *routing-instance-name* routing-options multipath],
[edit routing-instances *routing-instance-name* routing-options multipath]

Release Information Statement introduced in Junos OS Release 10.1.

Description Specify to have the algorithm that is used to determine the active path compare the AS numbers in the AS path. In a VPN scenario with multiple BGP paths, the algorithm selects as the active path the route whose AS numbers match. By default, the algorithm evaluates only the length and not the contents of the AS path.

Required Privilege Level routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation
- Configuring the Algorithm That Determines the Active Route to Evaluate AS Numbers in AS Paths for VPN Routes on page 208

classifiers

Syntax

```
classifiers {
  exp (classifier-name | default);
}
```

Hierarchy Level [edit class-of-service routing-instances *routing-instance-name*]

Release Information Statement introduced before Junos OS Release 7.4.

Description For routing instances with VRF table labels enabled, apply a custom MPLS EXP classifier to the routing instance. You can apply the default MPLS EXP classifier or one that is previously defined.

Default If you do not include this statement, the default MPLS EXP classifier is applied to the routing instance.

Options ***classifier-name***—Name of the behavior aggregate MPLS EXP classifier.

Required Privilege Level interface—To view this statement in the configuration.
interface-control—To add this statement to the configuration.

Related Documentation

- Applying Custom MPLS EXP Classifiers to Routing Instances in Layer 3 VPNs on page 190
- *Junos OS Network Interfaces Configuration Guide*

domain-id

Syntax domain-id *domain-id*;

Hierarchy Level [edit logical-systems *logical-system-name* routing-instances *routing-instance-name* protocols (ospf | ospf3)],
[edit routing-instances *routing-instance-name* protocols (ospf | ospf3)]

Release Information Statement introduced before Junos OS Release 7.4.

Description Specify a domain ID for a route. The domain ID identifies the OSPFv2 domain from which the route originated.

Default If the router ID is not configured in the routing instance, the router ID is derived from an interface address belonging to the routing instance.

Options ***domain-id***—IP address.

Required Privilege Level routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation

- Configuring an OSPF Domain ID on page 173

domain-vpn-tag

Syntax domain-vpn-tag *number*;

Hierarchy Level [edit logical-systems *logical-system-name* routing-instances *routing-instance-name* protocols (ospf | ospf3)],
[edit routing-instances *routing-instance-name* protocols (ospf | ospf3)]

Release Information Statement introduced before Junos OS Release 7.4.

Description Set a virtual private network (VPN) tag for OSPFv2 external routes generated by the provider edge (PE) router.

Options *number*—VPN tag.

Required Privilege Level routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation
- Configuring an OSPF Domain ID on page 173

dynamic-tunnels

Syntax

```
dynamic-tunnels tunnel-name {
  destination-networks prefix;
  source-address address;
  tunnel-type gre;
}
```

Hierarchy Level [edit logical-systems *logical-system-name* routing-options],
[edit routing-options]

Release Information Statement introduced before Junos OS Release 7.4.

Description Enable dynamic tunnel creation.

Options

destination-networks *prefix*—Specifies the IP version 4 (IPv4) prefix range for the destination network by including the **destination-networks** statement. Only tunnels within the specified IPv4 prefix range are allowed to be initiated.

source-address *address*—Specifies the source address for the generic routing encapsulation (GRE) tunnels. 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).

tunnel-name—Specifies the name of the dynamic tunnel.

tunnel-type gre—Specifies that a GRE tunnel is to be dynamically created.

Required Privilege Level routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation

- Configuring GRE Tunnels Dynamically on page 199
- *Junos OS Routing Protocols Configuration Guide*

independent-domain

Syntax independent-domain;

Hierarchy Level [edit logical-systems *logical-system-name* routing-instances *routing-instance-name* routing-options autonomous-system *autonomous-system*],
[edit routing-instances *routing-instance-name* routing-options autonomous-system *autonomous-system*],

Release Information Statement introduced before Junos OS Release 7.4.

Description Improve the transparency of Layer 3 VPN services for customer networks by preventing the IBGP routes that originate within an autonomous system (AS) in the customer network from being sent to a service provider's AS. Similarly, IBGP routes that originate within an AS in the service provider's network are prevented from being sent to a customer AS.

Required Privilege Level routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation
- Configuring Layer 3 VPNs to Carry IBGP Traffic on page 182
- Configuring Independent AS Domains

inet6-vpn

Syntax

```
inet6-vpn (any | multicast | unicast) {
  aggregate-label;
  prefix-limit maximum;
  rib-group rib-group-name;
}
```

Hierarchy Level

[edit logical-systems *logical-system-name* protocols bgp family],
[edit logical-systems *logical-system-name* protocols bgp group *group-name* family],
[edit protocols bgp family],
[edit protocols bgp group *group-name* family]

Release Information

Statement introduced before Junos OS Release 7.4.

Description

Enable IP version 6 (IPv6) on the provider edge (PE) router for the Layer 3 VPN.

Options

any—Configure the family type to be both multicast and unicast.

multicast—Configure the family type to be multicast. This means that the BGP peers are being used only to carry the unicast routes that are being used by multicast for resolving the multicast routes.

prefix-limit *maximum*—Maximum prefix limit.

Range: 1 through 4,294,967,295

Default: 1

rib-group *rib-group-name*—The name of the routing table group.

unicast—Configure the family type to be unicast. This means that the BGP peers only carry the unicast routes that are being used for unicast forwarding purposes.

Required Privilege Level

routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation

- Configuring Layer 3 VPNs to Carry IPv6 Traffic on page 178
- *Junos OS Routing Protocols Configuration Guide*

l3vpn-composite-nexthop

Syntax l3vpn-composite-nexthop;

Hierarchy Level [edit logical-systems *logical-system-name* routing-options],
[edit routing-options]

Release Information Statement introduced in Junos OS Release 9.5.
Statement introduced on T Series routers in Junos OS Release 10.4.

Description Accept larger numbers of Layer 3 VPN BGP updates with unique inner VPN labels. This feature is available on the M120 router, M320 router with Enhanced III FPC, MX Series router, and T Series routers. The neighboring PE routers are typically non–Juniper Networks routers configured to assign a unique inner label to each Layer 3 VPN BGP route.

Default This statement is disabled by default.

Required Privilege Level routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation

- Accepting BGP Updates with Unique Inner VPN Labels in Layer 3 VPNs on page 209

label

Syntax

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

Hierarchy Level

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

Release Information

Statement introduced in Junos OS Release 10.0.

Description

Specify label allocation and substitution policies on a per-route basis.

Options

allocation *label-allocation-policy*—Specify a policy to generate labels on a per-route basis.

substitution *label-substituion-policy*—Specify a policy to substitute labels on a per-route basis. The label substitution policy is used to determine whether or not a label should be substituted on an AS border router. The results of the policy operation are either accept (label substitution is performed) or reject (label substitution is not performed).

Default: accept

Required Privilege Level

routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation

- Configuring a Label Allocation and Substitution Policy for VPNs on page 191

maximum-paths

Syntax maximum-paths *path-limit* <log-interval *interval* | log-only | threshold *percentage*>;

Hierarchy Level [edit logical-systems *logical-system-name* routing-instances *routing-instance-name* routing-options],
[edit logical-systems *logical-system-name* routing-options],
[edit routing-instances *routing-instance-name* routing-options],
[edit routing-options]

Release Information Statement introduced in Junos OS Release 8.0.

Description Specify a maximum limit on the number of paths that can be installed into the routing tables. Using a path limit, you can curtail the number of paths received from a CE router in a VPN. Path limits apply only to dynamic routing protocols and are not applicable to static or interface routes.

Options ***path-limit***—Specify the maximum number of paths.

Range: 1 through 4,294,967,295 paths

log-interval—Minimum interval between log messages.

Range: 5 through 86,400 seconds

log-only—Generate warning messages only. No limit is placed on the number of paths stored in the routing tables.

threshold—Percentage of the path limit at which to begin sending warning log messages.

Required Privilege Level routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation

- Limiting the Number of Paths and Prefixes Accepted from CE Routers in Layer 3 VPNs on page 178

maximum-prefixes

Syntax maximum-prefixes *prefix-limit* <log-interval *interval* | log-only | threshold *percentage*>;

Hierarchy Level [edit logical-systems *logical-system-name* routing-instances *routing-instance-name* routing-options],
[edit logical-systems *logical-system-name* routing-options],
[edit routing-instances *routing-instance-name* routing-options],
[edit routing-options]

Release Information Statement introduced in Junos OS Release 8.0.

Description Specify a maximum limit on the number of prefixes that can be installed into the routing tables. Using a prefix limit, you can curtail the number of prefixes received from a CE router in a VPN. Prefix limits apply only to dynamic routing protocols and are not applicable to static or interface routes.

Options ***prefix-limit***—Specify the maximum number of prefixes.

Range: 1 through 4,294,967,295 prefixes

log-interval—Minimum interval between log messages.

Range: 5 through 86,400 seconds

log-only—Generate warning messages only. No limit is placed on the number of prefixes stored in the routing tables.

threshold—Percentage of the prefix limit at which to begin sending warning log messages.

Required Privilege Level routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation

- Limiting the Number of Paths and Prefixes Accepted from CE Routers in Layer 3 VPNs on page 178

metric

Syntax metric *number*;

Hierarchy Level [edit logical-systems *logical-system-name* routing-instances *routing-instance-name* protocols ospf area *area-id* sham-link-remote],
[edit routing-instances *routing-instance-name* protocols ospf area *area-id* sham-link-remote]

Release Information Statement introduced before Junos OS Release 7.4.

Description Specify the cost of using the OSPF sham link.

Default *number*—1

Options *number*—1 through 65,535

Required Privilege Level routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation
- Configuring OSPF Sham Links on page 172

multihop

Syntax multihop *ttl-value*;

Hierarchy Level
[edit logical-systems *logical-system-name* routing-instances *routing-instance-name* protocols bgp],
[edit logical-systems *logical-system-name* routing-instances *routing-instance-name* protocols bgp group *group-name*],
[edit logical-systems *logical-system-name* routing-instances *routing-instance-name* protocols bgp group *group-name* neighbor *address*],
[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*]

Release Information Statement introduced before Junos OS Release 7.4.

Description Configure an EBGP multihop session between the PE and customer edge (CE) routers of a Layer 3 VPN. This allows you to have one or more routers between the PE and CE routers.

Options ***ttl-value***—Specify the time-to-live (TTL) value for the multihop session to prevent routing loops.

Required Privilege Level
routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation
- Configuring EBGP Multihop Sessions Between PE and CE Routers in Layer 3 VPNs on page 182

multipath

Syntax

```
multipath {
    vpn-unequal-cost equal-external-internal;
    as-path-compare;
}
```

Hierarchy Level

[edit logical-systems *logical-system-name* routing-instances *routing-instance-name* routing-options],
[edit logical-systems *logical-system-name* routing-instances *routing-instance-name* routing-options rib *routing-table-name*],
[edit routing-instances *routing-instance-name* routing-options],
[edit routing-instances *routing-instance-name* routing-options rib *routing-table-name*]

Release Information

Statement introduced before Junos OS Release 7.4.
equal-external-internal option added for Junos OS Release 8.4.
as-path-compare option introduced in Junos OS Release 10.1

Description

Enable protocol-independent load balancing for Layer 3 VPNs. This allows the forwarding next hops for both the active route and alternative paths to be used for load balancing.

The options are explained separately.

Required Privilege Level

routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation

- Configuring Protocol-Independent Load Balancing in Layer 3 VPNs on page 206

no-vrf-propagate-ttl

Syntax no-vrf-propagate-ttl;

Hierarchy Level [edit logical-systems *logical-system-name* routing-instances *routing-instance-name*], [edit protocols routing-instances *routing-instance-name*]

Release Information Statement introduced in Junos OS Release 10.4.

Description Disable normal time-to-live (TTL) decrementing in a VRF routing instance. You configure this statement once per routing instance, and it affects only RSVP-signaled or LDP-signaled LSPs in the routing instance. When this router acts as an ingress router for an LSP, it pushes an MPLS header with a TTL value of 255, regardless of the IP packet TTL. When the router acts as the penultimate router, it pops the MPLS header without writing the MPLS TTL into the IP packet.

Default Normal TTL decrementing enabled; the TTL field value is decremented by 1 as the packet passes through each label-switched router in the LSP.

Required Privilege Level routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation

- Example: Disabling Normal TTL Decrementing in a VRF Routing Instance on page 319
- Disabling Normal TTL Decrementing in the *Junos OS MPLS Applications Configuration Guide*

routing-instances

Syntax

```
routing-instances routing-instance-name {
  classifiers {
    exp (classifier-name | default);
  }
}
```

Hierarchy Level [edit class-of-service]

Release Information Statement introduced before Junos OS Release 7.4.

Description For routing instances with the **vrf-table-label** statement in their configuration, apply a custom MPLS EXP classifier to the routing instance. You can apply the default MPLS EXP classifier or one that is previously defined.

Options ***routing-instance-name***—Name of the routing instance.

The other statements are explained separately.

Required Privilege Level routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation

- Applying Custom MPLS EXP Classifiers to Routing Instances in Layer 3 VPNs on page 190
- *Junos OS Network Interfaces Configuration Guide*

sham-link

Syntax

```
sham-link {
  local address;
}
```

Hierarchy Level [edit logical-systems *logical-system-name* routing-instances *routing-instance-name* protocols ospf],
[edit routing-instances *routing-instance-name* protocols ospf]

Release Information Statement introduced before Junos OS Release 7.4.

Description Configure a sham link for the Layer 3 VPN routing instance.

Options **local *address***—The address for the local endpoint of the sham link.

Required Privilege Level routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation

- Configuring OSPF Sham Links on page 172
- *Junos OS Routing Protocols Configuration Guide*

sham-link-remote

Syntax sham-link-remote *address* <metric *number*>;

Hierarchy Level [edit logical-systems *logical-system-name* routing-instances *routing-instance-name* protocols ospf area *area-id*],
[edit routing-instances *routing-instance-name* protocols ospf area *area-id*]

Release Information Statement introduced before Junos OS Release 7.4.

Description Configure the address for the remote end point of the sham link.

Options ***address***—Address for the remote end point of the sham link.

The other statement is explained separately.

Required Privilege Level routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation
- Configuring OSPF Sham Links for Layer 3 VPNs on page 171
- *Junos OS Routing Protocols Configuration Guide*

vpn-group-address

Syntax vpn-group-address *address*;

Hierarchy Level [edit logical-systems *logical-system-name* routing-instances *routing-instance-name* protocols pim],
[edit routing-instances *routing-instance-name* protocols pim]

Release Information Statement introduced before Junos OS Release 7.4.

Description Configure the group address for the Layer 3 VPN in the service provider's network.

Options ***address***—Address for the Layer 3 VPN in the service provider's network.

Required Privilege Level routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation
- Configuring Multicast Layer 3 VPNs on page 194
- *Junos OS Multicast Protocols Configuration Guide*

vpn-unequal-cost

Syntax

```
vpn-unequal-cost {
  equal-external-internal;
}
```

Hierarchy Level

[edit logical-systems *logical-system-name* routing-instances *routing-instance-name* routing-options multipath],
[edit logical-systems *logical-system-name* routing-instances *routing-instance-name* routing-options rib *routing-table-name* multipath],
[edit routing-instances *routing-instance-name* routing-options multipath],
[edit routing-instances *routing-instance-name* routing-options rib *routing-table-name* multipath]

Release Information

Statement introduced before Junos OS Release 7.4. The **equal-external-internal** option was added for Junos OS Release 8.4.

Description

Apply protocol-independent load balancing to VPN routes that are equal until their interior gateway protocol (IGP) metrics with regard to route selection. If you do not configure the **vpn-unequal-cost** statement, protocol-independent load balancing is applied to VPN routes that are equal until their router identifiers with regard to route selection.

Options

equal-external-internal—Specifies that both external and internal BGP paths can be selected for multipath.

Required Privilege Level

routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation

- Configuring Load Balancing for Layer 3 VPNs on page 206
- Load Balancing and IP Header Filtering for Layer 3 VPNs on page 191

vrf-propagate-ttl

Syntax vrf-propagate-ttl;

Hierarchy Level [edit logical-systems *logical-system-name* routing-instances *routing-instance-name*], [edit routing-instances *routing-instance-name*]

Release Information Statement introduced in Junos OS Release 10.4.

Description Enable normal time-to-live (TTL) decrementing in a VRF routing instance. You configure this statement once per routing instance, and it affects only RSVP-signaled or LDP-signaled LSPs in the routing instance. When this router acts as an ingress router for an LSP, it pushes an MPLS copies the TTL value from the IP packet header. When the router acts as the penultimate router, it pops the MPLS header and writes the MPLS TTL into the IP packet.

Default Normal TTL decrementing enabled; the TTL field value is decremented by 1 as the packet passes through each label-switched router in the LSP. This statement explicitly configures the default behavior for the VRF routing instance and is useful for overriding the **no-propagate-ttl** configured globally on the router at the **[edit protocols mpls]** hierarchy level.

Required Privilege Level routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation

- Example: Disabling Normal TTL Decrementing in a VRF Routing Instance on page 319
- Disabling Normal TTL Decrementing in the *Junos OS MPLS Applications Configuration Guide*

vrf-table-label

Syntax vrf-table-label;

Hierarchy Level [edit logical-systems *logical-system-name* routing-instances *routing-instance-name*], [edit routing-instances *routing-instance-name*]

Release Information Statement introduced before Junos OS Release 7.4.
Statement introduced in Junos OS Release 11.1 for EX Series switches.

Description Map the inner label of a packet to a specific VPN routing and forwarding (VRF) table. This allows the examination of the encapsulated IP header. This statement is not supported on 4 port E3 IQ PICs.

Required Privilege Level routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation

- Filtering Packets in Layer 3 VPNs Based on IP Headers on page 183
- Configuring EXP-Based Traffic Classification for VPLS on page 484
- Load Balancing and IP Header Filtering for Layer 3 VPNs on page 191

PART 4

Multicast VPNs

CHAPTER 15

Multicast VPNs Overview

This chapter provides an overview of multiprotocol BGP-based (MBGP) multicast VPNs (MVPNs), also known as next-generation multicast VPNs. For information about draft-rosen multicast VPNs, see the "Multicast over Layer 3 Draft-Rosen VPNs Overview" chapter in the *Junos Multicast Protocols Configuration Guide*. This chapter discusses the following topics:

MBGP Multicast VPN Sites

Multiprotocol BGP multicast VPNs (MBGP MVPNs) employ the intra-autonomous system (AS) next-generation BGP control plane and Protocol Independent Multicast (PIM) sparse mode as the data plane.

There are several multicast applications driving the deployment of next-generation Layer 3 MVPNs. Some of the key emerging applications include the following:

- Layer 3 VPN multicast service offered by service providers to enterprise customers
- Video transport applications for wholesale IPTV and multiple content providers attached to the same network
- Distribution of media-rich financial services or enterprise multicast services
- Multicast backhaul over a metro network

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.

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.

Multicast VPN Terminology

I

Inclusive tree — A single multicast distribution tree in the backbone that carries all the multicast traffic from a specified set of one or more multicast VPNs. An inclusive tree that carries the traffic of more than one multicast VPN is an aggregate inclusive tree. An inclusive tree contains as its members all the PE routers that attach to the receiver sites of any of the multicast VPNs using the tree.

S

Selective tree — A single multicast distribution tree in the backbone that carries traffic belonging only to a specified set of one or more multicast groups, from one or more multicast VPNs. An aggregate selective tree carries traffic for multicast groups that belong to different multicast VPNs. By default, traffic from most multicast groups could be carried by an inclusive tree, whereas traffic from high-bandwidth groups should be carried by a selective tree.

Multicast VPN Standards

MBGP MVPNs are defined in the following IETF Internet drafts:

- Internet draft draft-ietf-l3vpn-2547bis-mcast-bgp-03.txt, *BGP Encodings for Multicast in MPLS/BGP IP VPNs*
- Internet draft draft-ietf-l3vpn-2547bis-mcast-02.txt, *Multicast in MPLS/BGP IP VPNs*

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.

Related Documentation

- Example: Allowing MBGP MVPN Remote Sources
- Example: Configuring a PIM-SSM Provider Tunnel for an MBGP MVPN
- Example: Configuring MBGP Multicast VPN Extranets

CHAPTER 16

Configuring Multicast VPNs

This chapter describes how to configure multiprotocol BGP-based (MBGP) multicast VPNs (MVPNs), and discusses the following topics:

Introduction to Configuring MBGP MVPNs

You configure multiprotocol BGP-based (MBGP) multicast VPNs (MVPNs) at a number of different hierarchy levels within the Junos OS. However, a majority of MBGP MVPN statements are configured within a routing instance as follows:

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

```
protocols {
   mvpn {
      mvpn-mode (rpt-spt | spt-only);
      receiver-site;
      sender-site;
      route-target {
         export-target {
            target target-community;
            unicast;
         }
         import-target {
            target {
               target-value;
               receiver target-value;
               sender target-value;
            }
            unicast {
               receiver;
               sender;
            }
         }
      }
   }
}
provider-tunnel {
   pim-asm group-address address;
   pim-ssm {
      group-address address;
   }
   rsvp-te {
      label-switched-path-template {
         (default-template | lsp-template-name);
      }
      static-lsp lsp-name;
   }
   selective {
      group multicast--prefix/prefix-length {
         source ip--prefix/prefix-length {
            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 {
            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 {
          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 {
          pim-ssm {
            group-range multicast-prefix;
          }
          rsvp-te {
            label-switched-path-template {
              (default-template | lsp-template-name);
            }
            static-lsp lsp-name;
          }
          threshold-rate number;
        }
      }
    }
    threshold-rate number;
  }
```

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*]**

Related Documentation

- *Junos OS Multicast over Layer 3 VPNs Feature Guide*

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 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 6 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 master 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 blackholes 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 6 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.

Related Documentation

- Configuring Shared-Tree Data Distribution Across Provider Cores for Providers of MBGP MVPNs on page 394
- Using Wildcards to Configure Selective Point-to-Multipoint LSPs for an MBGP MVPN on page 407

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 master 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.

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.

Related Documentation

- Configuring SPT-Only Mode for Multiprotocol BGP-Based Multicast VPNs on page 392
- Using Wildcards to Configure Selective Point-to-Multipoint LSPs for an MBGP MVPN on page 407

Configuring VRF Route Targets for Routing Instances for an MBGP MVPN

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 on page 397
- Configuring the Import Target for an MBGP MVPN on page 397

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 on page 398
- Configuring the Import Target Unicast Parameters for an MBGP MVPN on page 398

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.

Related Documentation

- **family on page 423**
- **inet-mvpn on page 427**
- **inet6-mvpn on page 429**
- **vrf-advertise-selective on page 450**

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);
      }
    }
  }
}
```

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. A selective provider tunnel uses a point-to-multipoint LSP.

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 224.1.1.1/32 customer multicast group address, the 10.2.2.2/32 customer source address, and a virtual routing instance named **green**:

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 224.1.1.1/32 source 10.2.2.2/32 pim-ssm group-range 232.1.1.0/24

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 224.1.1.1/32 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.

Related Documentation

- Multicast VPN Terminology on page 386
- **pim-ssm on page 435**
- **group-range on page 425**
- **threshold-rate on page 446**

Configuring Point-to-Multipoint LSPs for an MBGP MVPN

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.

The following sections describe how to configure point-to-multipoint LSPs for MBGP MVPNs:

- Configuring Inclusive Point-to-Multipoint LSPs for an MBGP MVPN on page 403
- Configuring Selective Provider Tunnels for an MBGP MVPN on page 403

Configuring Inclusive Point-to-Multipoint LSPs for an MBGP MVPN

You can configure 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 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 {
    (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 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 {
            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 {
            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 {
            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 {
            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 on page 405
- Configuring the Multicast Source Address for an MBGP MVPN on page 405
- Configuring Static Selective Point-to-Multipoint LSPs for an MBGP MVPN on page 406
- Configuring Dynamic Selective Point-to-Multipoint LSPs for an MBGP MVPN on page 406
- Configuring the Threshold for Dynamic Selective Point-to-Multipoint LSPs for an MBGP MVPN on page 407
- Configuring the Tunnel Limit for Dynamic Selective Point-to-Multipoint LSPs for an MBGP MVPN on page 407

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]**

Using Wildcards to Configure Selective Point-to-Multipoint LSPs for an MBGP MVPN

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 on page 408
- Scenarios for Using Wildcard S-PMSI on page 409

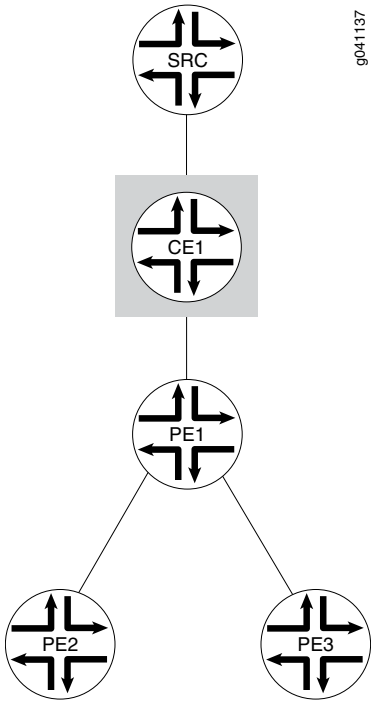
About S-PMSI

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 50 on page 409 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 50: 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 224.1.1.0/24 {
```

```
                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, 224.1.1.0/24) is bound to **sptnl3**. A join message for (*, 224.1.1.0/24) is bound to **sptnl2**.

Related Documentation

- Configuring a Selective Provider Tunnel Using Wildcards on page 413
- Example: Configuring Selective Provider Tunnels Using Wildcards on page 414
- Configuring SPT-Only Mode for Multiprotocol BGP-Based Multicast VPNs on page 392
- Configuring Shared-Tree Data Distribution Across Provider Cores for Providers of MBGP MVPNs on page 394

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 http://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 224.0.0/24 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 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 label-switched-path-template provider-tunnel2
```

6. Map the (*,224.0.0/24) join messages to a selective provider tunnel.

```
[edit routing-instances vpna provider-tunnel selective group 224.0.0/24
wildcard-source]
user@router# set rsvp-te label-switched-path-template provider-tunnel3
```

Related Documentation

- Using Wildcards to Configure Selective Point-to-Multipoint LSPs for an MBGP MVPN on page 407
- Example: Configuring Selective Provider Tunnels Using Wildcards on page 414
- Configuring SPT-Only Mode for Multiprotocol BGP-Based Multicast VPNs on page 392

- Configuring Shared-Tree Data Distribution Across Provider Cores for Providers of MBGP MVPNs on page 394

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 224.1.1.0/24 {
          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, 224.1.1.1) join message is bound to the sptnl3 S-PMSI autodiscovery route.
- A customer multicast (10.2.1.1, 224.1.1.1) join message is bound to the sptnl2 S-PMSI autodiscovery route.

- A customer multicast (10.1.1.1, 224.2.1.1) 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.

Related Documentation

- Using Wildcards to Configure Selective Point-to-Multipoint LSPs for an MBGP MVPN on page 407
- Configuring a Selective Provider Tunnel Using Wildcards on page 413

Tracing MBGP MVPN Traffic and Operations

To trace MBGP MVPN traffic, you can specify options with the **traceoptions** statement:

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

The following trace flags display the operations associated with multicast VPNs:

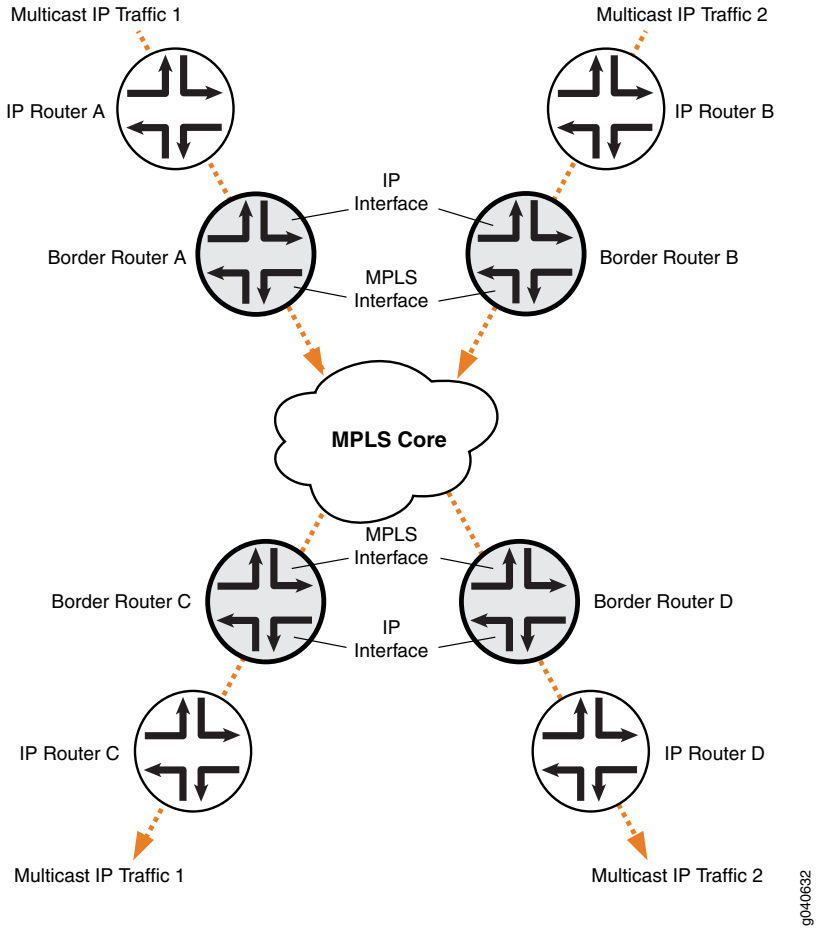
- **all**—All multicast VPN tracing options
- **error**—Error conditions
- **general**—General events
- **nlri**—Multicast VPN advertisements received or sent by means of BGP
- **normal**—Normal events
- **policy**—Policy processing
- **route**—Routing information
- **state**—State transitions
- **task**—Routing protocol task processing
- **timer**—Routing protocol timer processing
- **topology**—Multicast VPN topology changes caused by reconfiguration or advertisements received from other PE routers using BGP

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 nonforwarding 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 for BGP MPLS-based Internet multicast for control plane procedures. Support for ingress replication provider tunnels is also configured on all routers to form a full mesh of MPLS point-to-point label-switched paths (LSPs) for the data provider tunnel.

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. Internet multicast traffic is carried between the IP routers, through the MPLS cloud, using ingress replication provider 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 to associate PIM with the **mpls-internet-multicast** routing instance.

When an application requests to add a destination to the ingress replication provider tunnel, the resulting behavior differs depending on which mode has been configured for the tunnel:

- **existing-unicast-tunnel** —In this default mode, an existing unicast tunnel to the destination is used. If a unicast tunnel is not available, the destination is not added.
- **create-new-ucast-tunnel**—If this mode is configured, a new unicast tunnel to the destination is added to the ingress replication provider tunnel, and is deleted if the application requests to delete the destination.

The ingress replication provider tunnel can be selective or inclusive, matching the configuration of the provider tunnel in the routing instance.

To configure Internet multicast using ingress replication tunnels:

1. Configure the routing instance for MPLS Internet multicast for VPN-B.

```
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
232.1.1.1/32 source 192.168.195.145/32 ingress-replication
label-switched-path
```

5. Configure the MVPN protocol in the routing instance.

```
user@host# set routing-instances VPN-B protocols mvpn
```

6. Commit the configuration.

```
user@host# commit
```

7. Use the **show mvpn instance** ***instance-name*** command to verify that the instance has been created.

```
user@host# 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-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
   192.168.195.145/32:232.1.1.1/32 INGRESS-REPLICATION:MPLS Label
18:10.255.245.6         RM
```

Use this example to add the PIM protocol to your configuration.

1. Add the **mpls-internet-multicast** configuration statement under the **[protocols pim]** hierarchy level in the master instance.

   ```
   user@host# set protocols pim mpls-internet-multicast
   ```

2. Commit the configuration.

   ```
   user@host# commit
   ```

3. Use the **show ingress-replication mvpn** command to verify configuration settings.

   ```
   user@host> 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
   ```

4. Configure the ingress replication provider tunnel to be inclusive.

   ```
   user@host# set routing-instances VPN-A provider-tunnel ingress-replication
   create-new-ucast-tunnel set routing-instances VPN-A provider-tunnel
   ingress-replication label-switched-path label-switched-path-template
   default-template
   ```

5. Use the **show mvpn instance** *instance-name* command to verify that the tunnel is inclusive.

   ```
   user@host# show mvpn instance VPN-B

   user@host# run show mvpn instance

   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
  192.168.195.145/32:232.1.1.1/32 INGRESS-REPLICATION:MPLS Label
18:10.255.245.6         RM
```

CHAPTER 17

Summary of Multicast VPN Configuration Statements

The following sections explain the configuration statements that apply specifically to multiprotocol BGP-based (MBGP) multicast VPNs (MVPNs). The statements are arranged alphabetically.

create-new-ucast-tunnel

Syntax create-new-ucast-tunnel;

Hierarchy Level [edit routing-instances *routing-instance-name* provider-tunnel ingress-replication],
[edit routing-instances *routing-instance-name* provider-tunnel selective group *address* source *source-address* ingress-replication]

Release Information Statement introduced in Junos OS Release 10.4.

Description One of two modes for building unicast tunnels when ingress replication is configured for the provider tunnel. When this statement is configured, each time a new destination is added to the multicast distribution tree, a new unicast tunnel to the destination is created in the ingress replication tunnel. The new tunnel is deleted if the destination is no longer needed. Use this mode for RSVP LSPs using ingress replication.

Required Privilege Level routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation

- Configuring Ingress Replication for IP Multicast Using Next Gen MVPN on page 515
- Configuring Routing Instances for an MBGP MVPN on page 391
- **mpls-internet-multicast on page 432**
- **ingress-replication on page 430**
- **existing-unicast-tunnel on page 422**

existing-unicast-tunnel

Syntax existing-unicast-tunnel;

Hierarchy Level [edit routing-instances *routing-instance-name* provider-tunnel ingress-replication],
[edit routing-instances *routing-instance-name* provider-tunnel selective group *address* source *source-address* ingress-replication]

Release Information Statement introduced in Junos OS Release 10.4.

Description The default of two modes for selecting unicast tunnels when ingress replication is configured for the provider tunnel. When this is configured (or implied as default), each time a new destination is added to the multicast distribution tree, an existing unicast tunnel to the destination is used. If an existing tunnel is not available, the destination is not added. This is the only mode available when using LDP LSPs and ingress replication.

Required Privilege Level routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation

- Configuring Ingress Replication for IP Multicast Using Next Gen MVPN on page 515
- Configuring Routing Instances for an MBGP MVPN on page 391
- **mpls-internet-multicast on page 432**
- **ingress-replication on page 430**
- **existing-unicast-tunnel on page 422**

export-target

Syntax

```
export-target {
  target target-community;
  unicast;
}
```

Hierarchy Level

[edit logical-systems *logical-system-name* routing-instances *routing-instance-name* protocols mvpn route-target],
[edit routing-instances *routing-instance-name* protocols mvpn route-target]

Release Information

Statement introduced in Junos OS Release 8.4.

Description

Enable you to override the Layer 3 VPN import and export route targets used for importing and exporting routes for the MBGP MVPN network layer reachability information (NLRI).

Options

target *target-community*—Specify the export target community.

unicast—Use the same target community as specified for unicast.

Required Privilege Level

routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation

- Configuring the Export Target for an MBGP MVPN on page 397

family (VRF Advertisement)

Syntax

```
family {
  inet-mvpn;
  inet6-mvpn;
}
```

Hierarchy Level

[edit logical-systems *logical-system-name* routing-instances *routing-instance-name* **vrf-advertise-selective**],
[edit routing-instances *routing-instance-name* **vrf-advertise-selective**],

Release Information

Statement introduced in Junos OS Release 10.1.

Description

Explicitly enable IPv4 or IPv6 MVPN routes to be advertised from the VRF instance while preventing all other route types from being advertised.

The options are explained separately.

Required Privilege Level

routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation

- Configuring PIM-SSM GRE Selective Provider Tunnels on page 401
- **inet-mvpn on page 427**
- **inet6-mvpn on page 429**

group

Syntax

```
group address {
  source source-address {
    pim-ssm {
      group-range multicast-prefix;
    }
    rsvp-te {
      label-switched-path-template {
        (default-template | lsp-template-name);
      }
      static-lsp lsp-name;
    }
    threshold-rate number;
  }
}
```

Hierarchy Level

[edit logical-systems *logical-system-name* routing-instances *routing-instance-name* provider-tunnel selective],
[edit routing-instances *routing-instance-name* provider-tunnel selective]

Release Information

Statement introduced in Junos OS Release 8.5.

Description

Specify the IP address for the multicast group configured for point-to-multipoint label-switched paths (LSPs) and PIM-SSM GRE selective provider tunnels.

Options

address—Specify the IP address for the multicast group. This address must be a valid multicast group address.

The remaining statements are explained separately.

Required Privilege Level

routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation

- Configuring the Multicast Group Address for an MBGP MVPN on page 405
- Configuring PIM-SSM GRE Selective Provider Tunnels on page 401

group-range (MBGP MVPN Tunnel)

Syntax group-range *multicast-prefix*;

Hierarchy Level
[edit logical-systems *logical-system-name* routing-instances *routing-instance-name* provider-tunnel selective group *group-address* source *source-address* pim-ssm],
[edit logical-systems *logical-system-name* routing-instances *routing-instance-name* provider-tunnel selective group *group-address* wildcard-source pim-ssm],
[edit logical-systems *logical-system-name* routing-instances *routing-instance-name* provider-tunnel selective wildcard-group-inet wildcard-source pim-ssm],
[edit logical-systems *logical-system-name* routing-instances *routing-instance-name* provider-tunnel selective wildcard-group-inet6 wildcard-source pim-ssm],
[edit routing-instances *routing-instance-name* provider-tunnel selective group *group-address* source *source-address* pim-ssm],
[edit routing-instances *routing-instance-name* provider-tunnel selective group *group-address* wildcard-source pim-ssm],
[edit routing-instances *routing-instance-name* provider-tunnel selective wildcard-group-inet wildcard-source pim-ssm],
[edit routing-instances *routing-instance-name* provider-tunnel selective wildcard-group-inet6 wildcard-source pim-ssm]

Release Information Statement introduced in Junos OS Release 10.1.

Description Establish the multicast group address range to use for creating MBGP MVPN source-specific multicast selective PMSI tunnels.

Options ***multicast-prefix***—Multicast group address range to be used to create MBGP MVPN source-specific multicast selective PMSI tunnels.

Range: Any valid, nonreserved IPv4 multicast address range

Default: None

Required Privilege Level routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation
- Configuring PIM-SSM GRE Selective Provider Tunnels on page 401

import-target

Syntax

```
import-target {
  target {
    target-value;
    receiver target-value;
    sender target-value;
  }
  unicast {
    receiver;
    sender;
  }
}
```

Hierarchy Level [edit logical-systems *logical-system-name* routing-instances *routing-instance-name* protocols mvpn route-target],
[edit routing-instances *routing-instance-name* protocols mvpn route-target]

Release Information Statement introduced in Junos OS Release 8.4.

Description Enable you to override the Layer 3 VPN import and export route targets used for importing and exporting routes for the MBGP MVPN NLRI.

Options The remaining statements are explained separately.

Required Privilege Level routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation

- Configuring the Import Target for an MBGP MVPN on page 397

inet-mvpn (BGP)

Syntax

```
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);
      }
    }
  }
}
```

Hierarchy Level [edit logical-systems *logical-system-name* protocols bgp family],
[edit protocols bgp family],
[edit logical-systems *logical-system-name* protocols bgp group *group-name* family],
[edit protocols bgp group *group-name* family]

Release Information Statement introduced in Junos OS Release 8.4.

Description Enable the **inet-mvpn** address family in BGP.

Required Privilege Level routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation
- Configuring NLRI Parameters for an MBGP MVPN on page 400

inet-mvpn (VRF Advertisement)

Syntax

```
inet-mvpn;
```

Hierarchy Level [edit logical-systems *logical-system-name* routing-instances *routing-instance-name* **vrf-advertise-selective family**],
[edit routing-instances *routing-instance-name* **vrf-advertise-selective family**]

Release Information Statement introduced in Junos OS Release 10.1.

Description Enable IPv4 MVPN routes to be advertised from the VRF instance.

Required Privilege Level routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation
- Limiting Routes to Be Advertised by an MVPN VRF Instance on page 399

inet6-mvpn (BGP)

Syntax

```
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);
      }
    }
  }
}
```

Hierarchy Level

[edit logical-systems *logical-system-name* protocols bgp family],
[edit protocols bgp family],
[edit logical-systems *logical-system-name* protocols bgp group *group-name* family],
[edit protocols bgp group *group-name* family]

Release Information

Statement introduced in Junos OS Release 10.0.

Description

Enable the **inet6-mvpn** address family in BGP.

Required Privilege Level

routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation

- Configuring NLRI Parameters for an MBGP MVPN on page 400
- BGP Configuration Guidelines in the *Junos OS Routing Protocols Configuration Guide*

inet6-mvpn (VRF Advertisement)

Syntax inet6-mvpn;

Hierarchy Level [edit logical-systems *logical-system-name* routing-instances *routing-instance-name* **vrf-advertise-selective family**],
[edit routing-instances *routing-instance-name* **vrf-advertise-selective family**],

Release Information Statement introduced in Junos OS Release 10.1.

Description Enable IPv6 MVPN routes to be advertised from the VRF instance.

Required Privilege Level routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation
- Limiting Routes to Be Advertised by an MVPN VRF Instance on page 399

ingress-replication

Syntax

```
ingress-replication {
  (create-new-ucast-tunnel | existing-unicast-tunnel);
    label-switched-path-template {
    }
  }
```

Hierarchy Level [edit routing-instances *routing-instance-name* provider-tunnel],
[edit routing-instances *routing-instance-name* provider-tunnel selective group *address* source *source-address*]

Release Information Statement introduced in Junos OS Release 10.4.

Description A provider tunnel type used for passing multicast traffic between routers through the MPLS cloud, or between PE routers when using MVPN. The ingress replication provider tunnel uses MPLS point-to-point LSPs to create the multicast distribution tree.

Options

existing-unicast-tunnel—An existing tunnel to the destination is used for ingress replication. If an existing tunnel is not available, the destination is not added. Default mode if no option is specified.

create-new-ucast-tunnel—When specified, a new unicast tunnel to the destination is created and used for ingress replication. The unicast tunnel is deleted later if the destination is no longer included in the multicast distribution tree.

Required Privilege Level routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation

- Configuring Ingress Replication for IP Multicast Using Next Gen MVPN on page 515
- Configuring Routing Instances for an MBGP MVPN on page 391
- **create-new-ucast-tunnel on page 421**
- **existing-unicast-tunnel on page 422**
- **mpls-internet-multicast on page 432**

label-switched-path-template

Syntax

```
label-switched-path-template {
    (default-template | lsp-template-name);
}
```

Hierarchy Level

[edit logical-systems *logical-system-name* routing-instances *routing-instance-name* provider-tunnel rsvp-te],
[edit logical-systems *logical-system-name* routing-instances *routing-instance-name* provider-tunnel selective group *address* source *source-address* rsvp-te],
[edit routing-instances *routing-instance-name* provider-tunnel rsvp-te],
[edit routing-instances *routing-instance-name* provider-tunnel selective group *address* source *source-address* rsvp-te]

Release Information

Statement introduced in Junos OS Release 8.5.

Description

Specify the LSP template used for the point-to-multipoint LSP. There is no default setting for the **label-switched-path-template** statement, so you must configure either the default template using the **default-template** option or you must specify the name of your preconfigured point-to-multipoint LSP template.

Options

default-template—Specify that the default template be used for the point-to-multipoint LSP.

lsp-template-name—Specify the name of an LSP template to be used for the point-to-multipoint LSP.

Required Privilege Level

routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation

- Configuring Inclusive Point-to-Multipoint LSPs for an MBGP MVPN on page 403
- Configuring Dynamic Selective Point-to-Multipoint LSPs for an MBGP MVPN on page 406

mpls-internet-multicast

Syntax mpls-internet-multicast;

Hierarchy Level [edit routing-instances *routing-instance-name* instance-type]
[edit protocols pim]

Release Information Statement introduced in Junos OS Release 11.1.

Description A nonforwarding routing instance type that supports Internet multicast over an MPLS network for the default master instance. No interfaces can be configured for it. Only one **mpls-internet-multicast** instance can be configured for each logical system.

The **mpls-internet-multicast** configuration statement is also explicitly required under PIM in the master instance.

Required Privilege Level routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation

- Configuring Ingress Replication for IP Multicast Using Next Gen MVPN on page 515
- **ingress-replication on page 430**

mvpn

Syntax

```
mvpn {
  mvpn-mode (rpt-spt | spt-only);
  receiver-site;
  sender-site;
  route-target {
    export-target {
      target target-community;
      unicast;
    }
    import-target {
      target {
        target-value;
        receiver target-value;
        sender target-value;
      }
      unicast {
        receiver;
        sender;
      }
    }
  }
}
```

Hierarchy Level [edit logical-systems *logical-system-name* routing-instances *routing-instance-name* protocols],
[edit routing-instances *routing-instance-name* protocols]

Release Information Statement introduced in Junos OS Release 8.4.

Description Enable next-generation multicast VPNs in a routing instance.

Options **receiver-site**—Allow sites with multicast receivers.

sender-site—Allow sites with multicast senders.

The remaining statements are explained separately.

Required Privilege Level routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation

- Configuring Routing Instances for an MBGP MVPN on page 391

mvpn-mode

Syntax mvpn-mode (**rpt-spt** | **spt-only**);

Hierarchy Level [edit logical-systems *profile-name* routing-instances *instance-name* protocols mvpn],
[edit routing-instances *instance-name* protocols mvpn]

Release Information Statement introduced in Junos OS Release 10.0.

Description Configure the mode for customer PIM (C-PIM) join messages.

The remaining statements are explained separately.

Default **spt-only**

Required Privilege Level routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation
- Configuring Shared-Tree Data Distribution Across Provider Cores for Providers of MBGP MVPNs on page 394
- Configuring SPT-Only Mode for Multiprotocol BGP-Based Multicast VPNs on page 392

pim-asm

Syntax
```
pim-asm {
  group-address address;
}
```

Hierarchy Level [edit logical-systems *logical-system-name* routing-instances *routing-instance-name* provider-tunnel],
[edit routing-instances *routing-instance-name* provider-tunnel]

Release Information Statement introduced in Junos OS Release 8.3.

Description Specify a Protocol Independent Multicast (PIM) sparse mode provider tunnel for an MBGP MVPN.

Options **group-address *address***—PIM sparse mode provider tunnel group address.

Required Privilege Level routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation
- Configuring PIM Provider Tunnels for an MBGP MVPN on page 400

pim-ssm (Selective Tunnel)

Syntax

```
pim-ssm {
    group-range multicast-prefix;
}
```

Hierarchy Level

[edit logical-systems *logical-system-name* routing-instances *routing-instance-name* provider-tunnel selective group *group-address* source *source-address*],
[edit logical-systems *logical-system-name* routing-instances *routing-instance-name* provider-tunnel selective group *group-address* wildcard-source],
[edit logical-systems *logical-system-name* routing-instances *routing-instance-name* provider-tunnel selective wildcard-group-inet wildcard-source],
[edit logical-systems *logical-system-name* routing-instances *routing-instance-name* provider-tunnel selective wildcard-group-inet6 wildcard-source],
[edit routing-instances *routing-instance-name* provider-tunnel selective group *group-address* source *source-address*],
[edit routing-instances *routing-instance-name* provider-tunnel selective group *group-address* wildcard-source],
[edit routing-instances *routing-instance-name* provider-tunnel selective wildcard-group-inet wildcard-source],
[edit routing-instances *routing-instance-name* provider-tunnel selective wildcard-group-inet6 wildcard-source]

Release Information

Statement introduced in Junos OS Release 10.1.

Description

Establish the multicast group address range to use for creating MBGP MVPN source-specific multicast selective PMSI tunnels.

Required Privilege Level

routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation

- Configuring PIM-SSM GRE Selective Provider Tunnels on page 401

provider-tunnel

Syntax

```
provider-tunnel {
  ingress-replication {
    (create-new-ucast-tunnel | existing-unicast-tunnel);
      label-switched-path-template {
        (default-template | lsp-template-name);
      }
    }
  pim-asm group-address address;
  pim-ssm {
    group-address address;
  }
  rsvp-te {
    label-switched-path-template {
      (default-template | lsp-template-name);
    }
    static-lsp lsp-name;
  }
  selective {
    group multicast--prefix/prefix-length {
      source ip--prefix/prefix-length {
        ingress-replication {
        (create-new-ucast-tunnel | existing-unicast-tunnel);
          label-switched-path-template {
            (default-template | lsp-template-name);
          }
        }
        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 {
        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 {
          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 {
          pim-ssm {
            group-range multicast-prefix;
          }
          rsvp-te {
            label-switched-path-template {
              (default-template | lsp-template-name);
            }
            static-lsp lsp-name;
          }
          threshold-rate number;
        }
      }
    }
  }
```

Hierarchy Level [edit logical-systems *logical-system-name* routing-instances *routing-instance-name*], [edit routing-instances *routing-instance-name*]

Release Information Statement introduced in Junos OS Release 8.3.
The **selective** statement and substatements added in Junos OS Release 8.5.
The **ingress-replication** statement and substatements added in Junos OS Release 10.4.

Description Configure virtual private LAN service (VPLS) flooding of unknown unicast, broadcast, and multicast traffic using point-to-multipoint LSPs. Also configure point-to-multipoint LSPs for MBGP MVPNs.

Options The remaining statements are explained separately.

Required Privilege Level routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation

- Flooding Unknown Traffic Using Point-to-Multipoint LSPs on page 507
- Configuring Inclusive Point-to-Multipoint LSPs for an MBGP MVPN on page 403

route-target

Syntax

```
route-target {
  export-target {
    target target-community;
    unicast;
  }
  import-target {
    target {
      target-value;
      receiver target-value;
      sender target-value;
    }
    unicast {
      receiver;
      sender;
    }
  }
}
```

Hierarchy Level [edit logical-systems *logical-system-name* routing-instances *routing-instance-name* protocols mvpn],
[edit routing-instances *routing-instance-name* protocols mvpn]

Release Information Statement introduced in Junos OS Release 8.4.

Description Enable you to override the Layer 3 VPN import and export route targets used for importing and exporting routes for the MBGP MVPN NLRI.

Default The multicast VPN routing instance uses the import and export route targets configured for the Layer 3 VPN.

Options The remaining statements are explained separately.

Required Privilege Level routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation

- Configuring VRF Route Targets for Routing Instances for an MBGP MVPN on page 396

rpt-spt

Syntax rpt-spt;

Hierarchy Level [edit logical-systems *profile-name* routing-instances *instance-name* protocols mvpn mvpn-mode],
[edit routing-instances *instance-name* protocols mvpn mvpn-mode]

Release Information Statement introduced in Junos OS Release 10.0.

Description Use rendezvous-point trees for customer PIM (C-PIM) join messages, and switch to the shortest-path tree after the source is known.

Required Privilege Level routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation
- Configuring Shared-Tree Data Distribution Across Provider Cores for Providers of MBGP MVPNs on page 394

rsvp-te

Syntax

```
rsvp-te {
  label-switched-path-template {
    (default-template | lsp-template-name);
  }
  static-lsp lsp-name;
}
```

Hierarchy Level

[edit logical-systems *logical-system-name* routing-instances *routing-instance-name* provider-tunnel],
[edit logical-systems *logical-system-name* 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* wildcard-source],
[edit logical-systems *logical-system-name* routing-instances *routing-instance-name* provider-tunnel selective wildcard-group-inet wildcard-source],
[edit logical-systems *logical-system-name* routing-instances *routing-instance-name* provider-tunnel selective wildcard-group-inet6 wildcard-source],
[edit routing-instances *routing-instance-name* provider-tunnel],
[edit routing-instances *routing-instance-name* provider-tunnel selective group *address* source *source-address*]
[edit routing-instances *routing-instance-name* provider-tunnel selective group *address* wildcard-source],
[edit routing-instances *routing-instance-name* provider-tunnel selective wildcard-group-inet wildcard-source],
[edit routing-instances *routing-instance-name* provider-tunnel selective wildcard-group-inet6 wildcard-source]

Release Information

Statement introduced in Junos OS Release 8.5.

Description

Configure the properties of the RSVP traffic-engineered point-to-multipoint LSP for MBGP MVPNs.

The remaining statements are explained separately.

Required Privilege Level

routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation

- Configuring Selective Provider Tunnels for an MBGP MVPN on page 403

selective

Syntax

```
selective {
  group multicast-prefix/prefix-length {
    source ip-prefix/prefix-length {
      ingress-replication {
        (create-new-ucast-tunnel | existing-unicast-tunnel);
          label-switched-path-template {
            (default-template | lsp-template-name);
          }
        }
      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 {
        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 {
      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 {
      pim-ssm {
```

```
        group-range multicast-prefix;
      }
      rsvp-te {
        label-switched-path-template {
          (default-template | lsp-template-name);
        }
        static-lsp lsp-name;
      }
      threshold-rate number;
    }
  }
}
```

Hierarchy Level

[edit logical-systems *logical-system-name* routing-instances *routing-instance-name* provider-tunnel],
[edit routing-instances *routing-instance-name* provider-tunnel]

Release Information

Statement introduced in Junos OS Release 8.5.
The **ingress-replication** statement and substatements added in Junos OS Release 10.4.

Description

Configure selective point-to-multipoint LSPs for an MBGP MVPN. Selective point-to-multipoint LSPs send traffic only to the receivers configured for the MBGP MVPNs, helping to minimize flooding in the service provider's network.

The remaining statements are explained separately.

Required Privilege Level

routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation

- Configuring Selective Provider Tunnels for an MBGP MVPN on page 403
- Configuring PIM-SSM GRE Selective Provider Tunnels on page 401

source

Syntax

```
source source-address {
  pim-ssm {
    group-range multicast-prefix;
  }
  rsvp-te {
    label-switched-path-template {
      (default-template | lsp-template-name);
    }
    static-lsp lsp-name;
  }
  threshold-rate number;
}
```

Hierarchy Level

[edit logical-systems *logical-system-name* routing-instances *routing-instance-name* provider-tunnel selective group *address*],
[edit routing-instances *routing-instance-name* provider-tunnel selective group *address*]

Release Information

Statement introduced in Junos OS Release 8.5.

Description

Specify the IP address for the multicast source. This statement is a part of the point-to-multipoint LSP and PIM-SSM GRE selective provider tunnel configuration for MBGP MVPNs.

Options

source-address—IP address for the multicast source.

The remaining statements are explained separately.

Required Privilege Level

routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation

- Configuring the Multicast Source Address for an MBGP MVPN on page 405
- Configuring PIM-SSM GRE Selective Provider Tunnels on page 401

spt-only

Syntax spt-only;

Hierarchy Level [edit logical-systems *profile-name* routing-instances *instance-name* protocols mvpn mvpn-mode],
[edit routing-instances *instance-name* protocols mvpn mvpn-mode]

Release Information Statement introduced in Junos OS Release 10.0.

Description Set the MVPN mode to learn about active multicast sources using multicast VPN source-active routes. This is the default mode.

Required Privilege Level routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation
- Configuring SPT-Only Mode for Multiprotocol BGP-Based Multicast VPNs on page 392

static-lsp

Syntax static-lsp *lsp-name*;

Hierarchy Level [edit logical-systems *logical-system-name* routing-instances *routing-instance-name* provider-tunnel rsvp-te],
[edit logical-systems *logical-system-name* routing-instances *routing-instance-name* provider-tunnel selective group *address* source *source-address* rsvp-te],
[edit routing-instances *routing-instance-name* provider-tunnel rsvp-te],
[edit routing-instances *routing-instance-name* provider-tunnel selective group *address* source *source-address* rsvp-te]

Release Information Statement introduced in Junos OS Release 8.5.

Description Specify the name of the static point-to-multipoint LSP used for an MBGP MVPN. Use this statement to specify the static LSP for both inclusive and selective point-to-multipoint LSPs.

Required Privilege Level routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation
- Configuring Selective Provider Tunnels for an MBGP MVPN on page 403

target

Syntax

```
target target-value {
    receiver target-value;
    sender target-value;
}
```

Hierarchy Level

[edit logical-systems *logical-system-name* routing-instances *routing-instance-name* protocols mvpn route-target import-target],
[edit routing-instances *routing-instance-name* protocols mvpn route-target import-target]

Release Information

Statement introduced in Junos OS Release 8.4.

Description

Specify the target value when importing sender and receiver site routes.

Options

target-value—Specify the target value when importing sender and receiver site routes.

receiver—Specify the target community used when importing receiver site routes.

sender—Specify the target community used when importing sender site routes.

Required Privilege Level

routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation

- Configuring the Import Target Receiver and Sender for an MBGP MVPN on page 398

threshold-rate

Syntax threshold-rate *kbps*;

Hierarchy Level
[edit logical-systems *logical-system-name* 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 *group-address* wildcard-source],
[edit logical-systems *logical-system-name* routing-instances *routing-instance-name* provider-tunnel selective wildcard-group-inet wildcard-source],
[edit logical-systems *logical-system-name* routing-instances *routing-instance-name* provider-tunnel selective wildcard-group-inet6 wildcard-source],
[edit routing-instances *routing-instance-name* provider-tunnel selective group *address* source *source-address*]
[edit routing-instances *routing-instance-name* provider-tunnel selective group *address* wildcard-source]
[edit routing-instances *routing-instance-name* provider-tunnel selective wildcard-group-inet wildcard-source],
[edit routing-instances *routing-instance-name* provider-tunnel selective wildcard-group-inet6 wildcard-source]

Release Information Statement introduced in Junos OS Release 8.5.

Description Specify the data threshold required before a new tunnel is created for a dynamic selective point-to-multipoint LSP. This statement is part of the configuration for point-to-multipoint LSPs for MBGP MVPNs and PIM-SSM GRE or RSVP-TE selective provider tunnels.

Options *number*—Specify the data threshold required before a new tunnel is created.

Range: 0 through 1,000,000 kilobits per second. Specifying 0 is equivalent to not including the statement.

Required Privilege Level
routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation

- Configuring the Threshold for Dynamic Selective Point-to-Multipoint LSPs for an MBGP MVPN on page 407
- Configuring PIM-SSM GRE Selective Provider Tunnels on page 401
- Configuring Intra-AS Selective Provider Tunnels

traceoptions

Syntax

```
traceoptions {
    file filename <files number> <size size> <world-readable | no-world-readable>;
    flag flag <flag-modifier> <disable>;
}
```

Hierarchy Level

[edit logical-systems *logical-system-name* routing-instances *routing-instance-name* protocols mvpn],
[edit routing-instances *routing-instance-name* protocols mvpn]

Release Information

Statement introduced in Junos OS Release 8.4.

Description

Trace traffic flowing through an MBGP MVPN.

Options

disable—(Optional) Disable the tracing operation. You can use this option to disable a single operation when you have defined a broad group of tracing operations, such as **all**.

file *filename*—Name of the file to receive the output of the tracing operation. Enclose the name in quotation marks (" ").

files *number*—(Optional) Maximum number of trace files. When a trace file named ***trace-file*** reaches this size, it is renamed ***trace-file*.0**. When ***trace-file*** again reaches its maximum size, ***trace-file*.0** is renamed ***trace-file*.1** and ***trace-file*** is renamed ***trace-file*.0**. This renaming scheme continues until the maximum number of trace files is reached. Then the oldest trace file is overwritten.

If you specify a maximum number of files, you also must specify a maximum file size with the **size** option.

Range: 2 through 1000 files

Default: 2 files

flag *flag*—Tracing operation to perform. To specify more than one tracing operation, include multiple **flag** statements. You can specify any of the following flags:

- **all**—All multicast VPN tracing options
- **error**—Error conditions
- **general**—General events
- **nlri**—Multicast VPN advertisements received or sent by means of the BGP
- **normal**—Normal events
- **policy**—Policy processing
- **route**—Routing information
- **state**—State transitions
- **task**—Routing protocol task processing
- **timer**—Routing protocol timer processing

- **topology**—Multicast VPN topology changes caused by reconfiguration or advertisements received from other provider edge (PE) routers using BGP

flag-modifier—(Optional) Modifier for the tracing flag. You can specify the following modifiers:

- **detail**—Provide detailed trace information
- **disable**—Disable the tracing flag
- **receive**—Trace received packets
- **send**—Trace sent packets

no-world-readable—Do not allow any user to read the log file.

size *size*—(Optional) Maximum size of each trace file, in kilobytes (KB), megabytes (MB), or gigabytes (GB). When a trace file named ***trace-file*** reaches this size, it is renamed ***trace-file*.0**. When ***trace-file*** again reaches its maximum size, ***trace-file*.0** is renamed ***trace-file*.1** and ***trace-file*** is renamed ***trace-file*.0**. This renaming scheme continues until the maximum number of trace files is reached. Then the oldest trace file is overwritten.

If you specify a maximum file size, you also must specify a maximum number of trace files with the **files** option.

Syntax: ***x*k** to specify kilobytes, ***x*m** to specify megabytes, or ***x*g** to specify gigabytes

Range: 10 KB through the maximum file size supported on your system

Default: 1 MB

world-readable—Allow any user to read the log file.

Required Privilege Level

routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation

- Tracing MBGP MVPN Traffic and Operations on page 415

tunnel-limit

Syntax tunnel-limit *number*;

Hierarchy Level [edit logical-systems *logical-system-name* routing-instances *routing-instance-name* provider-tunnel selective],
[edit routing-instances *routing-instance-name* provider-tunnel selective]

Release Information Statement introduced in Junos OS Release 8.5.

Description Specify a limit on the number of tunnels that can be created for a point-to-multipoint LSP.

Options ***number***—Specify the tunnel limit.

Range: 0 through 1024

Required Privilege Level routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation
- Configuring the Tunnel Limit for Dynamic Selective Point-to-Multipoint LSPs for an MBGP MVPN on page 407

unicast

Syntax
```
unicast {
  receiver;
  sender;
}
```

Hierarchy Level [edit logical-systems *logical-system-name* routing-instances *routing-instance-name* protocols mvpn route-target import-target],
[edit routing-instances *routing-instance-name* protocols mvpn route-target import-target]

Release Information Statement introduced in Junos OS Release 8.4.

Description Specify the same target community configured for unicast.

Options **receiver**—Specify the unicast target community used when importing receiver site routes.

sender—Specify the unicast target community used when importing sender site routes.

Required Privilege Level routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation
- Configuring the Import Target Unicast Parameters for an MBGP MVPN on page 398

vrf-advertise-selective

Syntax

```
vrf-advertise-selective {
  family {
    inet-mvpn;
    inet6-mvpn;
  }
}
```

Hierarchy Level [edit logical-systems *logical-system-name* routing-instances *routing-instance-name*], [edit routing-instances *routing-instance-name*]

Release Information Statement introduced in Junos OS Release 10.1.

Description Explicitly enable IPv4 or IPv6 MVPN routes to be advertised from the VRF instance while preventing all other route types from being advertised.

The options are explained separately.

Required Privilege Level routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation

- Limiting Routes to Be Advertised by an MVPN VRF Instance on page 399

wildcard-group-inet

Syntax

```
wildcard-group-inet {
  wildcard-source {
    pim-ssm {
      group-range multicast-prefix;
    }
    rsvp-te {
      label-switched-path-template {
        (default-template | lsp-template-name);
      }
      static-lsp lsp-name;
    }
    threshold-rate number;
  }
}
```

Hierarchy Level

[edit logical-systems *logical-system-name* routing-instances *routing-instance-name* provider-tunnel selective],
[edit routing-instances *routing-instance-name* provider-tunnel selective]

Release Information

Statement introduced in Junos OS Release 10.0.

Description

Configure a wildcard group matching any group IPv4 address.

The remaining statements are explained separately.

Required Privilege Level

routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation

- **wildcard-group-inet6 on page 452**
- Example: Configuring Selective Provider Tunnels Using Wildcards on page 414
- Using Wildcards to Configure Selective Point-to-Multipoint LSPs for an MBGP MVPN on page 407
- Configuring a Selective Provider Tunnel Using Wildcards on page 413

wildcard-group-inet6

Syntax

```
wildcard-group-inet6 {
  wildcard-source {
    pim-ssm {
      group-range multicast-prefix;
    }
    rsvp-te {
      label-switched-path-template {
        (default-template | lsp-template-name);
      }
      static-lsp lsp-name;
    }
    threshold-rate number;
  }
}
```

Hierarchy Level

[edit logical-systems *logical-system-name* routing-instances *routing-instance-name* provider-tunnel selective],
[edit routing-instances *routing-instance-name* provider-tunnel selective]

Release Information

Statement introduced in Junos OS Release 10.0.

Description

Configure a wildcard group matching any group IPv6 address.

The remaining statements are explained separately.

Required Privilege Level

routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation

- **wildcard-group-inet on page 451**
- Example: Configuring Selective Provider Tunnels Using Wildcards on page 414
- Using Wildcards to Configure Selective Point-to-Multipoint LSPs for an MBGP MVPN on page 407
- Configuring a Selective Provider Tunnel Using Wildcards on page 413

wildcard-source

Syntax

```
wildcard-source {
  pim-ssm {
    group-range multicast-prefix;
  }
  rsvp-te {
    label-switched-path-template {
      (default-template | lsp-template-name);
    }
    static-lsp lsp-name;
  }
```

Hierarchy Level

[edit logical-systems *logical-system-name* routing-instances *routing-instance-name* provider-tunnel selective group *group-prefix*],
[edit logical-systems *logical-system-name* routing-instances *routing-instance-name* provider-tunnel selective wildcard-group-inet],
[edit logical-systems *logical-system-name* routing-instances *routing-instance-name* provider-tunnel selective wildcard-group-inet6],
[edit routing-instances *routing-instance-name* provider-tunnel selective group *group-prefix*],
[edit logical-systems *logical-system-name* routing-instances *routing-instance-name* provider-tunnel selective wildcard-group-inet],
[edit logical-systems *logical-system-name* routing-instances *routing-instance-name* provider-tunnel selective wildcard-group-inet6],
[edit routing-instances *routing-instance-name* provider-tunnel selective wildcard-group-inet],
[edit routing-instances *routing-instance-name* provider-tunnel selective wildcard-group-inet6]

Release Information

Statement introduced in Junos OS Release 10.0.

Description

Configure a selective provider tunnel for a shared tree using a wildcard source.

The remaining statements are explained separately.

Required Privilege Level

routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation

- **wildcard-group-inet on page 451**
- **wildcard-group-inet6 on page 452**
- Example: Configuring Selective Provider Tunnels Using Wildcards on page 414
- Using Wildcards to Configure Selective Point-to-Multipoint LSPs for an MBGP MVPN on page 407
- Configuring a Selective Provider Tunnel Using Wildcards on page 413

PART 5

VPLS

CHAPTER 18

VPLS Overview

This chapter provides an overview of virtual private LAN service (VPLS) as it is implemented in the Junos OS.

For information about virtual private networks (VPNs) and the differences between Layer 2 VPNs, Layer 3 VPNs, and VPLS, see “VPN Overview” on page 3.

This chapter discusses the following topics that provide background information about VPLS:

Introduction to VPLS

VPLS is an Ethernet-based point-to-multipoint Layer 2 VPN. It allows you to connect geographically dispersed Ethernet local area networks (LAN) sites to each other across an MPLS backbone. For 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, in its implementation and configuration, has much in common with a Layer 2 VPN. In VPLS, a packet originating within a service provider customer’s network is sent first to a customer edge (CE) device (for example, a router or Ethernet switch). It is then sent to a provider edge (PE) router within the service provider’s network. The packet traverses the service provider’s network over a MPLS label-switched path (LSP). It arrives at the egress PE router, which then forwards the traffic to the CE device at the destination customer site.

The difference is that for VPLS, packets can traverse the service provider’s network in point-to-multipoint fashion, meaning that a packet originating from a CE device can be

broadcast to all the PE routers participating in a VPLS routing instance. In contrast, a Layer 2 VPN forwards packets in point-to-point fashion only.

The paths carrying VPLS traffic between each PE router participating in a routing instance are called pseudowires. The pseudowires are signaled using either BGP or LDP.

Supported Platforms and PICs

Virtual private LAN service (VPLS) is supported on all M Series routers except the M160.

VPLS is supported on all J Series, MX Series, and T Series routers.

VPLS is supported on the following SRX Services Gateways for the branch:

- SRX100
- SRX210
- SRX240
- SRX650

VPLS is supported on the following PICs:

- All ATM2 IQ PICs
- 4-port Fast Ethernet PIC with 10/100 Base-TX interfaces PIC
- 1-port, 2-port, and 10-port Gigabit Ethernet PICs
- 1-port, 2-port, and 4-port Gigabit Ethernet PICs with SFP
- 1-port 10-Gigabit Ethernet PIC
- 1-port and 2-port Gigabit Ethernet Intelligent Queuing (IQ) PICs
- 4-port and 8-port Gigabit Ethernet IQ2 PICs with SFP
- 1-port 10-Gigabit Ethernet IQ2 PIC with XFP
- 4-port, quad-wide Gigabit Ethernet PIC
- 10-port 10-Gigabit OSE PIC

VPLS Routing and Virtual Ports

Because VPLS carries Ethernet traffic across a service provider network, it must mimic an Ethernet network in some ways. When a PE router configured with a VPLS routing instance receives a packet from a CE device, it first determines whether it has the destination of the VPLS packet in the appropriate routing table. If it does, it forwards the packet to the appropriate PE router or CE device. If it does not, it broadcasts the packet to all other PE routers and CE devices that are members of that VPLS routing instance. In both cases, the CE device receiving the packet must be different from the one sending the packet.

When a PE router receives a packet from another PE router, it first determines whether it has the destination of the VPLS packet in the appropriate routing table. If it does, the

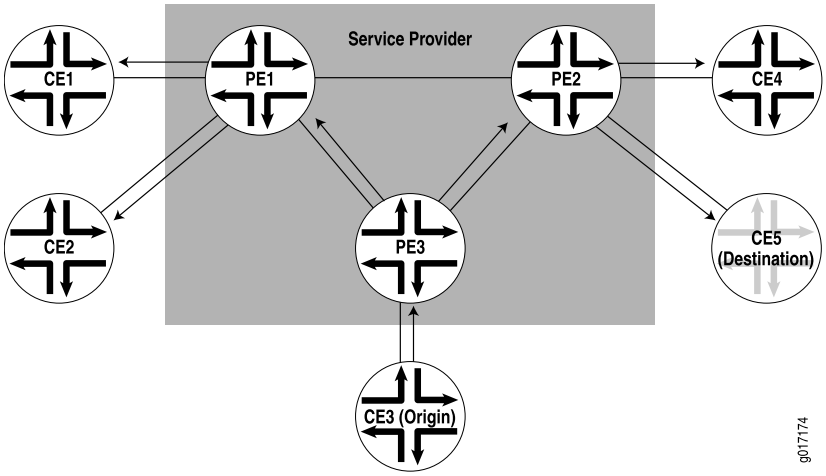
PE router either forwards the packet or drops it depending on whether the destination is a local or remote CE device:

- If the destination is a local CE device, the PE router forwards the packet to it.
- If the destination is a remote CE device (connected to another PE router), the PE router discards the packet.

If the PE router cannot determine the destination of the VPLS packet, it floods the packet to all attached CE devices.

This process is illustrated in Figure 51 on page 459.

Figure 51: Flooding a Packet with an Unknown Destination to All PE Routers in the VPLS Instance



VPLS can be directly connected to an Ethernet switch. Layer 2 information gathered by an Ethernet switch (for example, media access control [MAC] addresses and interface ports) is included in the VPLS routing instance table. However, instead of all VPLS interfaces being physical switch ports, the router allows remote traffic for a VPLS instance to be delivered across an MPLS LSP and arrive on a virtual port. The virtual port emulates a local, physical port. Traffic can be learned, forwarded, or flooded to the virtual port in almost the same way as traffic is sent to a local port.

The VPLS routing table learns MAC address and interface information for both physical and virtual ports. The main difference between a physical port and a virtual port is that the router captures additional information from the virtual port, an outgoing MPLS label used to reach the remote site and an incoming MPLS label for VPLS traffic received from the remote site. The virtual port is generated dynamically on a Tunnel Services Physical Interface Card (PIC) when you configure VPLS on the router.

You can also configure VPLS without a Tunnel Services PIC. To do so, you use a label-switched interface (LSI) to provide VPLS functionality. An LSI MPLS label is used as the inner label for VPLS. This label maps to a VPLS routing instance. On the PE router, the LSI label is stripped and then mapped to a logical LSI interface. The Layer 2 Ethernet frame is then forwarded using the LSI interface to the correct VPLS routing instance.

One restriction on flooding behavior in VPLS is that traffic received from remote PE routers is never forwarded to other PE routers. This restriction helps prevent loops in the core network. However, if a CE Ethernet switch has two or more connections to the same PE router, you must enable the Spanning Tree Protocol (STP) on the CE switch to prevent loops. STP is supported on MX Series routers only.

The Junos OS allows standard Bridge Protocol Data Unit (BPDU) frames to pass through emulated Layer 2 connections, such as those configured with Layer 2 VPNs, Layer 2 circuits, and VPLS routing instances. However, CE Ethernet switches that generate proprietary BPDU frames might not be able to run STP across Juniper Networks routing platforms configured for these emulated Layer 2 connections.

NOTE: Under certain circumstances, VPLS provider routers might duplicate an Internet Control Message Protocol (ICMP) reply from a CE router when a PE router has to flood an ICMP request because the destination MAC address has not yet been learned. The duplicate ICMP reply can be triggered when a CE router with promiscuous mode enabled is connected to a PE router. The PE router automatically floods the promiscuous mode–enabled CE router, which then returns the ICMP request to the VPLS provider routers. The VPLS provider routers consider the ICMP request to be new and flood the request again, creating a duplicate ping reply.

VPLS and Aggregated Ethernet Interfaces

You can configure aggregated Ethernet interfaces between CE devices and PE routers for VPLS routing instances. Traffic is load-balanced across all of the links in the aggregated interface. If one or more links in the aggregated interface fails, the traffic is switched to the remaining links.

Forwarding is based on a lookup of the DA MAC address. For the remote site, if a packet needs to be forwarded over an LSP, the packet is encapsulated and forwarded through the LSP. If the packet destination is a local site, it is forwarded over appropriate local site interface. For an aggregated Ethernet interface on the local site, packets are sent out of the load-balanced child interface. The Packet Forwarding Engine acquires the child link to transmit the data.

When a received packet does not have a match to a MAC address in the forwarding database, the packet is forwarded over a set of interfaces determined from a lookup in the flooding database based on the incoming interface. This is denoted by a flood next hop. The flood next hop can include the aggregated Ethernet interface as the set of interfaces to flood the packet.

Each VPLS routing instance configured on a PE router has its own forwarding database entries that associate all of the MAC addresses the VPLS routing instance acquires with each corresponding port. A route is added to the kernel with a MAC address as the prefix and the next hop used to reach the destination. The route is an interface if the destination is local. For a remote destination, the route is a next hop for the remote site.

For local aggregated Ethernet interfaces on M Series and T Series routers, learning is based on the parent aggregated Ethernet logical interface. To age out MAC addresses for aggregated Ethernet interfaces, each Packet Forwarding Engine is queried to determine where the individual child interfaces are located. MAC addresses are aged out based on the age of the original interface.

For MX Series routers, when a Dense Port Concentrator (DPC) learns a MAC address it causes the Routing Engine to age out the entry. This behavior applies to all logical interfaces. For an aggregated Ethernet logical interface, once all the member DPCs have aged out the entry, the entry is deleted from the Routing Engine.

For information about how to configure aggregated Ethernet interfaces for VPLS routing instances, see “Configuring Aggregated Ethernet Interfaces for VPLS” on page 488.

VPLS Multihoming

VPLS multihoming allows you to connect a customer site to multiple PE routers to provide redundant connectivity while preventing the formation of Layer 2 loops in the service provider’s network. A VPLS site multihomed to two or more PE routers provides redundant connectivity in the event of a PE router-to-CE device link failure or the failure of a PE router.

When multihoming a VPLS site (potentially in different autonomous systems [ASs]), the PE routers connected to the same site can either be configured with the same VPLS edge (VE) device identifier or with different VE device identifiers. In the latter case, you must run STP on the CE device, and possibly on the PE routers, to construct a loop-free VPLS topology.

If the PE routers are connected to the same site and assigned the same VE device identifier, a loop-free topology is constructed using a routing mechanism such as BGP path selection. When a BGP speaker receives two equivalent network layer reachability information (NLRI) advertisements, it applies standard path selection criteria such as local preference and AS path length to determine which NLRI to choose; it selects only one.

Because a PE router picks one of the received NLRI advertisements with a particular VE device identifier, it establishes pseudowires to only one of the remote PE routers, the PE router that originated the winning advertisement. This prevents multiple paths from being created in the network between sites, preventing the formation of Layer 2 loops in the network. If the selected PE router fails, all PE routers in the network automatically switch to the backup PE router and establish pseudowires through the backup PE router.

Two VPLS NLRIs are considered equivalent from a path selection perspective if the following are the same:

- Route distinguisher
- VE device identifier
- VE block offset

If two PE routers are assigned the same VE device identifier in a given VPLS, they must also advertise the same VE block size for a given VE offset. The PE routers can be configured with the same route distinguisher or with distinct route distinguishers.

We recommend that you configure distinct route distinguishers for each multihomed router. Configuring distinct route distinguishers helps with faster convergence when the connection to a primary router goes down. It also requires the other PE routers to maintain additional state information.

NOTE: Traffic loss can occur when the old pseudowires are brought down and new ones established.

Interoperability between BGP Signaling and LDP Signaling in VPLS

You can configure a VPLS routing instance where some of the PE routers use BGP for signaling and some use LDP for signaling.

The following concepts form the basis of the configuration needed to include both BGP-signaled and LDP-signaled PE routers in a VPLS routing instance:

- PE router mesh group—Consists of a set of routers participating in a VPLS routing instance that share the same signaling protocol, either BGP or LDP, and are also fully meshed. Each VPLS routing instance can have just one BGP mesh group. However, you can configure multiple LDP mesh groups for each routing instance.
- Border router—A PE router that must be reachable by all of the other PE routers participating in a VPLS routing instance, whether they are LDP-signaled or BGP-signaled. Bidirectional pseudowires are created between the border router and all of these PE routers. The border router is aware of the composition of each PE mesh group configured as a part of the VPLS routing instance. It can also have direct connections to local CE routers, allowing it to act as a typical PE router in a VPLS routing instance.

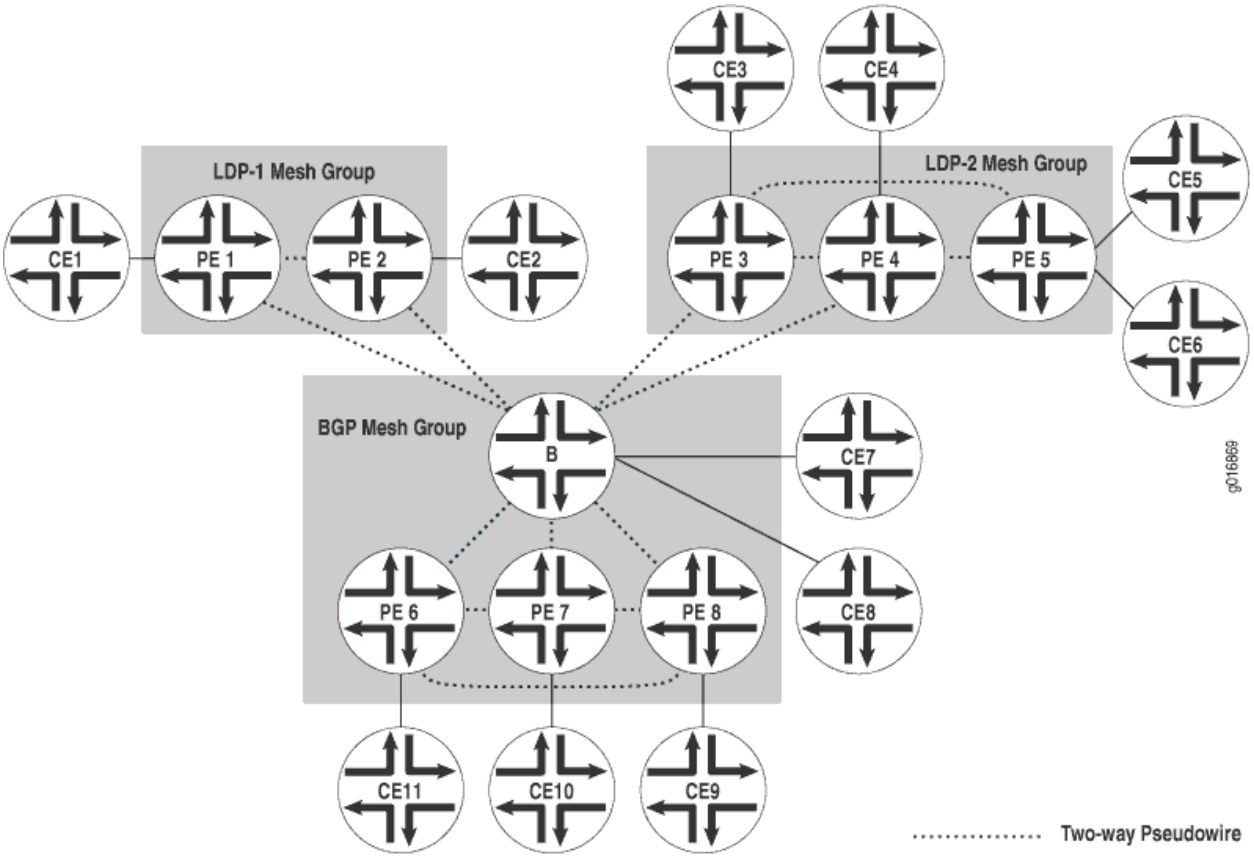
The following sections describe how the LDP-signaled and BGP-signaled PE routers function when configured to interoperate within a VPLS routing instance:

- LDP-Signaled and BGP-Signaled PE Router Topology on page 462
- Flooding Unknown Packets Across Mesh Groups on page 464
- Unicast Packet Forwarding on page 464

LDP-Signaled and BGP-Signaled PE Router Topology

Figure 52 on page 463 illustrates a topology for a VPLS routing instance configured to support both BGP and LDP signaling. Router B is the border router. Routers PE1 and PE2 are in the LDP-signaled mesh group LDP-1. Routers PE3, PE4, and PE5 are in the LDP-signaled mesh group LDP-2. Routers PE6, PE7, PE8, and router B (the border router) are in the BGP-signaled mesh group. The border router also acts as a standard VPLS PE router (having local connections to CE routers). All of the PE routers shown are within the same VPLS routing instance.

Figure 52: BGP and LDP Signaling for a VPLS Routing Instance



Two-way pseudowires are established between the PE routers in each mesh group and between each PE router in the VPLS routing instance and the border router. In Figure 52 on page 463, two-way pseudowires are established between routers PE1 and PE2 in mesh group LDP-1, routers PE3, PE4, and PE5 in mesh group LDP-2, and routers PE6, PE7, and PE8 in the BGP mesh group. Routers PE1 through PE8 also all have two-way pseudowires to the Border router. Based on this topology, the LDP-signaled routers are able to interoperate with the BGP-signaled routers. Both the LDP-signaled and BGP-signaled PE routers can logically function within a single VPLS routing instance.

NOTE: The following features are not supported for VPLS routing instances configured with both BGP and LDP signaling:

- Point-to-multipoint LSPs
- Integrated routing and bridging
- IGMP snooping

Flooding Unknown Packets Across Mesh Groups

Broadcast, multicast, and unicast packets of unknown origin received from a PE router are flooded to all local CE routers. They are also flooded to all of the PE routers in the VPLS routing instance except the PE routers that are a part of the originating PE router mesh group.

For example, if a multicast packet is received by the border router in Figure 52 on page 463, it is flooded to the two local CE routers. It is also flooded to routers PE1 and PE2 in the LDP-1 mesh group and to routers PE3, PE4, and PE5 in the LDP-2 mesh group. However, the packet is not flooded to routers PE6, PE7, and PE8 in the BGP mesh group.

Unicast Packet Forwarding

The PE border router is made aware of the composition of each PE router mesh group. From the data plane, each PE router mesh group is viewed as a virtual pseudowire LAN. The border router is configured to interconnect all of the PE router mesh groups belonging to a single VPLS routing instance. To interconnect the mesh groups, a common MAC table is created on the border router.

Unicast packets originating within a mesh group are dropped if the destination is another PE router within the same mesh group. However, if the destination MAC address of the unicast packet is a PE router located in a different mesh group, the packet is forwarded to that PE router.

VPLS Label Blocks Operation

A virtual private LAN service (VPLS) is a Layer 2 (L2) service that emulates a local area network (LAN) across a wide area network (WAN). VPLS labels are defined and exchanged in the Border Gateway Protocol (BGP) control plane. In the Junos OS implementation, label blocks are allocated and used in the VPLS control plane for two primary functions: autodiscovery and signaling.

- Autodiscovery—A method for automatically recognizing each provider edge (PE) router in a particular VPLS domain, using BGP update messages.
- Signaling—Each pair of PE routers in a VPLS domain sends and withdraws VPN labels to each other. The labels are used to establish and dismantle pseudowires between the routers. Signaling is also used to transmit certain characteristics of a pseudowire.

The PE router uses BGP extended communities to identify the members of its VPLS. Once the PE router discovers its members, it is able to establish and tear down pseudowires between members by exchanging and withdrawing labels and transmitting certain characteristics of the pseudowires.

The PE router sends common update messages to all remote PE routers, using a distinct BGP update message, thereby reducing the control plane load. This is achieved by using VPLS label blocks.

Elements of Network Layer Reachability Information

VPLS BGP network layer reachability information (NLRI) is used to exchange VPLS membership and parameters. The elements of a VPLS BGP NLRI are defined in Table 9 on page 465.

Table 9: NLRI Elements

Element	Acronym	Description	Default Size (Octets)
Length		Total length of the NLRI size represented in bytes.	2
Route Distinguisher	RD	Unique identifier for each routing instance configured on a PE.	8
VPLS Edge ID	VE ID	Unique number to identify the edge site.	2
VE Block Offset	VBO	Value used to identify a label block from which a label value is selected to set up pseudowires for a remote site.	2
VE Block Size	VBS	Indicates the number of pseudowires that peers can have in a single block.	2
Label Base	LB	Starting value of the label in the advertised label block.	3

Requirements for NLRI Elements

Junos OS requires a unique route distinguisher (RD) for each routing instance configured on a PE router. A PE router might use the same RD across a VPLS (or VPN) domain or it might use different RDs. Using different RDs helps identify the originator of the VPLS NLRI.

The VPLS edge (VE) ID can be a unique VE ID, site ID, or customer edge (CE) ID. The VE ID is used by a VPLS PE router to index into label blocks used to derive the transmit and receive VPN labels needed for transport of VPLS traffic. The VE ID identifies a particular site, so it needs to be unique within the VPLS domain, except for some scenarios such as multihoming.

All PE routers have full mesh connectivity with each other to exchange labels and set up pseudowires. The VE block size (VBS) is a configurable value that represents the number of label blocks required to cover all the pseudowires for the remote peer.

A single label block contains 8 labels (1 octet) by default. The default VBS in Junos OS is 2 blocks (2 octets) for a total of 16 labels.

How Labels are Used in Label Blocks

Each PE router creates a mapping of the labels in the label block to the sites in a VPLS domain. A PE router advertising a label block with a block offset indicates which sites can use the labels to reach it. When a PE router is ready to advertise its membership to

a VPLS domain, it allocates a label block and advertises the VPLS NLRI. In this way, other PE routers in the same VPLS domain can learn of the existence of the VPLS and set up pseudowires to it if needed. The VPLS NLRI advertised for this purpose is referred to as the *default VPLS NLRI*. The label block in the default VPLS NLRI is referred to as the *default label block*.

Label Block Composition

A label block (set of labels) is used to reach a given site ID. A single label block contains 8 labels (1 octet) by default. The VBS is 2 octets by default in Junos OS.

The label block advertised is defined as a label base (LB) and a VE block size (VBS). It is a contiguous set of labels (LB, LB+1,...LB+VBS-1). For example, when Router PE-A sends a VPLS update, it sends the same label block information to all other PE routers. Each PE router that receives the LB advertisement infers the label intended for Router PE-A by adding its own site ID to the label base.

In this manner, each receiving PE gets a unique label for PE-A for that VPLS. This simple method is enhanced by using a VE block offset (VBO).

A label block is defined as: <Label Base (LB), VE block offset (VBO), VE block size (VBS)> is the set {LB+VBO, LB+VBO+1,...,LB+VBO+VBS-1}.

Label Blocks in Junos OS

Instead of a single large label block to cover all VE IDs in a VPLS, the Junos OS implementation contains several label blocks, each with a different label base. This makes label block management easier, and also allows Router PE-A to seemlessly integrate a PE router joining a VPLS with a site ID not covered by the set of label blocks that Router PE-A has already advertised.

VPLS Label Block Structure

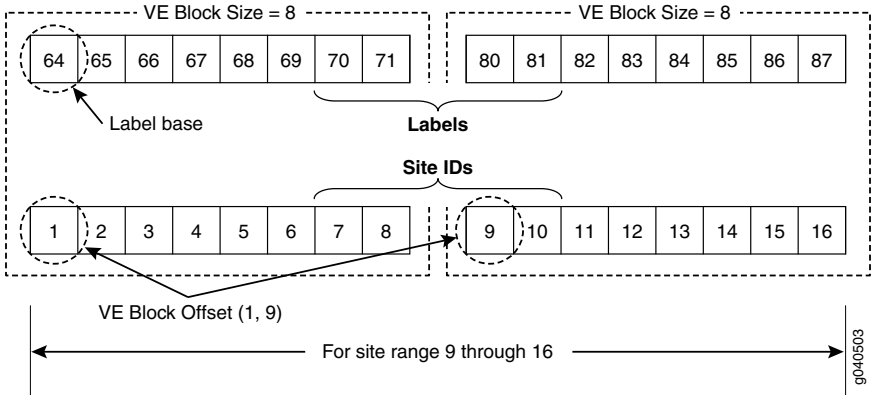
This section illustrates how a label block is uniquely identified.

A VPLS BGP NLRI with site ID V, VE block offset VBO, VE block size VBS, and label base LB communicates the following to its peers:

- Label block for V: Labels from LB to (LB + VBS -1).
- Remote VE set for V: from VBO to (VBO + VBS -1).

The label block advertised is a set of labels used to reach a given site ID. If there are several label blocks, the remote VE set helps to identify which label block to use. The example in Figure 53 on page 467 illustrates label blocks. There are two blocks and each block has eight labels. In this example, the label values are 64 to 71 and 80 to 87.

Figure 53: VPLS Label Block Structure



To create a one-to-one mapping of these 16 labels to 16 sites, assume the site IDs are the numbers 1 to 16, as shown in the illustration. The site block indicates which site ID can use which label in the label block. So, in the first block, site ID 1 uses 64, site ID 2 uses 65, and so forth. Finally, site ID 8 uses 71. The 9th site ID will use the second block instead of the first block.

The labels are calculated by comparing the values of VBO < = Local site ID < (VBO +VBS). Consequently, site ID 9 uses 80, site ID 10 uses 81, and so on.

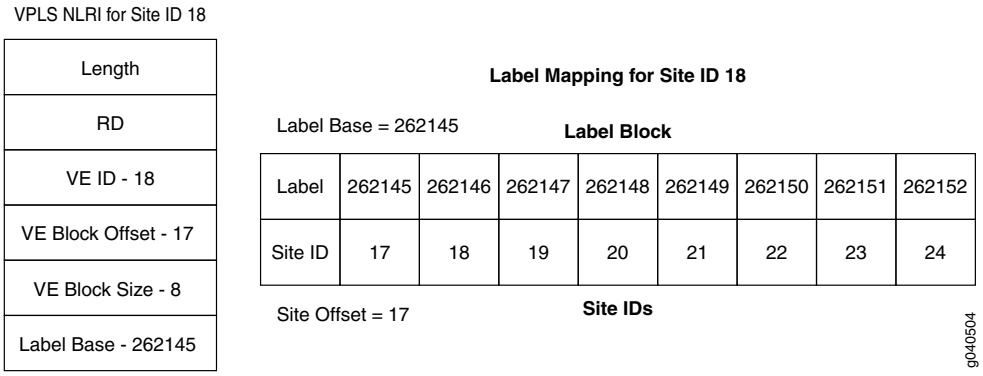
To further illustrate the one-to-one mapping of labels to sites, assume a label block with site offset of 1 and a label base of 10. The combination of label base and block offset contained in the VPLS NLRI provides the mapping of labels to site IDs. The block offset is the starting site ID that can use the label block as advertised in the VPLS NLRI.

To advertise the default VPLS NLRI, a PE router picks a starting block offset that fits its own site ID and is such that the end block offset is a multiple of a single label block. In Junos OS a single label block is eight labels by default.

The end block offset is the last site ID that maps to the last label in the label block. The end offset for the first block is 8 which maps to label 17 and the second block is 16. For example, a site with ID 3 picks a block offset of 1 and advertises a label block of size 8 to cover sites with IDs 1 to 8. A site with ID 10 picks a block offset of 9 to cover sites with IDs 9 to 16.

The VPLS NLRI shown in Figure 54 on page 468 is for site ID 18. The label base contains value 262145. The block offset contains value 17. The illustration shows which site IDs correspond to which labels.

Figure 54: Label Mapping Example



If a PE router configured with site ID 17 is in the same VPLS domain as a PE router configured with site ID 18, it receives the VPLS NLRI as shown in Figure 3. So it uses label 262145 to send traffic to site 18. Similarly, a PE router configured with site ID 19 uses label 262147 to send traffic to a PE router configured with site ID 18. However, only PE routers configured with site IDs 17 to 24 can use the label block shown to set up pseudowires.

Related Documentation

- Example: Building a VPLS From Router 1 to Router 3 on page 523

PE Router Mesh Groups for VPLS Routing Instances

A PE router mesh group consists of a set of routers participating in a VPLS routing instance that share the same signaling protocol, either BGP or LDP. Each VPLS routing instance can have just one BGP mesh group. However, you can configure multiple LDP mesh groups for each routing instance.

The JUNOS Software can support up to 16 mesh groups on MX Series routers and up to 128 on M Series and T Series routers. However, two mesh groups are created by default, one for the CE routers and one for the PE routers. Therefore, the maximum number of user-defined mesh groups is 14 for MX Series routers and 126 for M Series and T Series routers. PE router mesh groups are not supported on J Series routers.

The following describes the default behavior of mesh groups in regards to BGP-signaled PE routers discovered automatically and LDP-signaled PE routers configured statically:

- BGP-signaled PE routers—Automatically discovered PE routers that use BGP for signaling are associated with the default VE mesh group. You cannot configure the Junos OS to associate these routers with a user-defined VE mesh group.
- LDP-signaled PE routers—PE routers statically configured using forwarding equivalence class (FEC)-128 LDP signaling are placed in a default mesh group. However, you can configure a VE mesh group and associate each LDP FEC-128 neighbor with it. Each configured VE mesh group contains a set of VEs that are in the same interior gateway protocol (IGP) routing instance and are fully meshed with each other in the control and data planes.

VPLS Standards

VPLS is described in the following Internet draft and RFC:

- Internet draft draft-ietf-l2vpn-vpls-bgp-08.txt, *Virtual Private LAN Service (VPLS) Using BGP for Auto-discovery and Signaling* (expires December 2006).
- RFC 4762 (FEC 128, control bit **0**, and Ethernet pseudowire type hexadecimal **0x0005**), *Virtual Private LAN Service (VPLS) Using Label Distribution Protocol (LDP) Signaling*.

You can access Internet RFCs and drafts on the IETF website at **http://www.ietf.org**.

CHAPTER 19

Configuring VPLS

This chapter describes how to configure VPLS, discussing the following topics:

Introduction to Configuring VPLS

Virtual private LAN service (VPLS) allows you to provide a point-to-multipoint LAN between a set of sites in a virtual private network (VPN).

To configure VPLS functionality, you must enable VPLS support on the provider edge (PE) router. You must also configure PE routers to distribute routing information to the other PE routers in the VPLS and configure the circuits between the PE routers and the customer edge (CE) routers.

Each VPLS is configured under a routing instance of type **vpls**. A **vpls** routing instance can transparently carry Ethernet traffic across the service provider's network. As with other routing instances, all logical interfaces belonging to a VPLS routing instance are listed under that instance.

To configure VPLS, include the following statements:

```
description text;
forwarding-options {
   family vpls {
      filter input input-filter-name;
      flood input flood-filter-name;
   }
   fast-reroute-priority (high | medium | low);
}
instance-type vpls;
interface interface-name;
route-distinguisher (as-number:id | ip-address:id);
vrf-export [ policy-names ];
vrf-import [ policy-names ];
vrf-target target:target-id;
protocols {
   vpls {
      active-interface {
         any;
         primary interface-name;
      }
      connectivity-type (ce | irb);
      interface-mac-limit limit;
      label-block-size size;
      mac-table-aging-time time;
      mac-table-size size;
      neighbor neighbor-id;
      no-tunnel-services;
      site site-name {
         active-interface {
            any;
            primary interface-name;
         }
         interface interface-name {
            interface-mac-limit limit;
         }
         multi-homing;
```

```
        site-identifier identifier;
        site-preference preference-value;
      }
      site-range number;
      traceoptions {
        file filename <files number> <size size> <world-readable | no-world-readable>;
        flag flag <flag-modifier> <disable>;
      }
      tunnel-services {
        devices device-names;
        primary primary-device-name;
      }
      vpls-id vpls-id;
    }
  }
  provider-tunnel {
    rsvp-te {
      label-switched-path-template {
        (default-template | lsp-template-name);
      }
      static-lsp lsp-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***]**

For VPLS, only some of the statements in the **[edit routing-instances]** hierarchy are valid. For the full hierarchy, see the *Junos OS Routing Protocols Configuration Guide*.

In addition to these statements, you must configure MPLS label-switched paths (LSPs) between the PE routers, IBGP sessions between the PE routers, and an interior gateway protocol (IGP) on the PE and provider (P) routers.

By default, VPLS is disabled.

Many configuration procedures for VPLS are identical to the procedures for Layer 2 VPNs and Layer 3 VPNs.

Configuring VPLS Routing Instances

To configure a VPLS routing instance, include the **vpls** statement:

```
vpls {
  active-interface {
    any;
    primary interface-name;
  }
  connectivity-type (ce | irb | permanent);
  interface-mac-limit limit;
  label-block-size size;
  mac-table-aging-time time;
```

```
    mac-table-size size;
    neighbor neighbor-id;
    no-tunnel-services;
    site site-name {
      active-interface {
        any;
        primary interface-name;
      }
      interface interface-name {
        interface-mac-limit limit;
      }
      multi-homing;
      site-identifier identifier;
      site-preference preference-value;
    }
    site-range number;
    traceoptions {
      file filename <files number> <size size> <world-readable | no-world-readable>;
      flag flag <flag-modifier> <disable>;
    }
    tunnel-services {
      devices device-names;
      primary primary-device-name;
    }
    vpls-id vpls-id;
  }
```

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: You cannot configure a routing protocol (OSPF, RIP, IS-IS or BGP) inside a VPLS routing instance (**instance-type vpls**). The Junos CLI disallows this configuration.

The configuration for the VPLS routing instance statements is explained in the following sections:

Configuring BGP Signaling for VPLS

You can configure BGP signaling for the VPLS routing instance. BGP is used to signal the pseudowires linking each of the PE routers participating in the VPLS routing instance. The pseudowires carry VPLS traffic across the service provider's network between the VPLS sites.

NOTE: You cannot configure both BGP signaling and LDP signaling for the same VPLS routing instance. If you attempt to configure the statements that enable BGP signaling for the VPLS routing instance (the **site**, **site-identifier**, and **site-range** statements) and the statements that enable LDP signaling for the same instance (the **neighbor** and **vpls-id** statements), the commit operation fails.

Configure BGP signaling for the VPLS routing instance by completing the steps in the following sections:

- Configuring the VPLS Site Name and Site Identifier on page 475
- Configuring Automatic Site Identifiers for VPLS on page 476
- Configuring the Site Range on page 476
- Configuring the VPLS Site Interfaces on page 477
- Configuring the VPLS Site Preference on page 477

Configuring the VPLS Site Name and Site Identifier

When you configure BGP signaling for the VPLS routing instance, on each PE router you must configure each VPLS site that has a connection to the PE router. All the Layer 2 circuits provisioned for a VPLS site are listed as the set of logical interfaces (using the **interface** statement) within the **site** statement.

You must configure a site name and site identifier for each VPLS site.

To configure the site name and the site identifier, include the **site** and the **site-identifier** statements:

```
site site-name {
    interface interface-name {
        interface-mac-limit limit;
    }
    site-identifier identifier;
}
```

The numerical identifier can be any number from 1 through 65,534 that uniquely identifies the local VPLS site.

You can include these statements at the following hierarchy levels:

- **[edit routing-instances *routing-instance-name* protocols vpls]**

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

Configuring Automatic Site Identifiers for VPLS

When you enable automatic site identifiers, the Junos OS automatically assigns site identifiers to VPLS sites. To configure automatic site identifiers for a VPLS routing instance, include the **automatic-site-id** statement:

```
automatic-site-id {
  collision-detect-time seconds;
  new-site-wait-time seconds;
  reclaim-wait-time minimum seconds maximum seconds;
  startup-wait-time seconds;
}
```

You can include this statement at the following hierarchy levels:

- **[edit routing-instances *routing-instance-name* protocols vpls site *site-name*]**
- **[edit logical-systems *logical-system-name* routing-instances *routing-instance-name* protocols vpls site *site-name*]**

The **automatic-site-id** statement includes a number of options that control different delays in network layer reachability information (NLRI) advertisements. All of these options are configured with default values. See the statement summary for the **automatic-site-id** statement for more information.

The **automatic-site-id** statement includes the following options:

- **collision-detect-time**—The time in seconds to wait after a claim advertisement is sent to the other routers in a VPLS instance before a PE router can begin using a site identifier. If the PE router receives a competing claim advertisement for the same site identifier during this time period, it initiates the collision resolution procedure for site identifiers.
- **new-site-wait-time**—The time in seconds to wait to receive VPLS information for a newly configured routing instance or a new site. This time interval is also applied whenever the automatic site identifier feature is activated on a VPLS routing instance other than at startup. Effectively, this timer indicates how long to wait before an attempt is made to allocate a site identifier. This timer is also triggered whenever a VPLS routing instance is enabled.
- **reclaim-wait-time**—The time to wait before attempting to claim a site identifier after a collision. A collision occurs whenever an attempt is made to claim a site identifier by two separate VPLS sites.
- **startup-wait-time**—The time in seconds to wait at startup to receive all the VPLS information for the route targets configured on the other PE routers included in the VPLS routing instance.

Configuring the Site Range

When you enable BGP signaling for each VPLS routing instance, you need to configure a site range. The site range specifies the total number of sites in the VPLS. To configure a site range, include the **site-range** statement:

site-range *number*;

You can include this statement at the following hierarchy levels:

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

NOTE: When you configure the site range, you need to specify a value that is greater than the site identifier. The following configuration example illustrates this issue:

```
protocols {
   vpls {
      site-range 20;
      no-tunnel-services;
      site sample {
         site-identifier 3;
      }
   }
}
```

This configuration is valid. However, if the site identifier was a value greater than 20, the VPLS connection would fail.

Configuring the VPLS Site Interfaces

All the Layer 2 circuits you configure for a VPLS site are listed as a set of logical interfaces within the VPLS site configuration.

To configure a logical interface for the VPLS site, include the **interface** statement:

```
interface interface-name {
   interface-mac-limit limit;
}
```

You can include these statements at the following hierarchy levels:

- **[edit routing-instances** *routing-instance-name* **protocols vpls site** *site-name***]**
- **[edit logical-systems** *logical-system-name* **routing-instances** *routing-instance-name* **protocols vpls]**

You can also configure a limit on the number of MAC addresses that can be learned from the specified interface. For more information, see “Limiting the Number of MAC Addresses Learned from an Interface” on page 481.

Configuring the VPLS Site Preference

You can specify the preference value advertised for a particular VPLS site. The site preference value is encoded in the BGP local preference attribute. When a PE router receives multiple advertisements with the same VPLS edge (VE) device identifier, the advertisement with the highest local preference value is preferred.

To configure the VPLS site preference, include the **site-preference** statement:

site-preference *preference-value*;

You can also specify either the **backup** option or the **primary** option for the **site-preference** statement. The backup option specifies the preference value as 1, the lowest possible value, ensuring that the VPLS site is the least likely to be selected. The primary option specifies the preference value as 65,535, the highest possible value, ensuring that the VPLS site is the most likely to be selected.

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

Configuring LDP Signaling for VPLS

You can configure LDP as the signaling protocol for a VPLS routing instance. This functionality is described in RFC 4762, *Virtual Private LAN Service (VPLS) Using Label Distribution Protocol (LDP) Signaling*.

The Junos OS does not support all of RFC 4762. When enabling LDP signaling for a VPLS routing instance, network engineers should be aware that only the following values are supported:

- FEC—**128** (supports only FEC-128)
- Control bit—**0**
- Ethernet pseudowire type—**0x0005** (hexadecimal)

To enable LDP signaling for the set of PE routers participating in the same VPLS routing instance, you need to use the **vpls-id** statement configured at the **[edit routing-instances *routing-instance-name* protocols vpls]** hierarchy level to configure the same VPLS identifier on each of the PE routers. The VPLS identifier must be globally unique. When each VPLS routing instance (domain) has a unique VPLS identifier, it is possible to configure multiple VPLS routing instances between a given pair of PE routers.

LDP signaling requires that you configure a full-mesh LDP session between the PE routers in the same VPLS routing instance. Neighboring PE routers are statically configured. Tunnels are created between the neighboring PE routers to aggregate traffic from one PE router to another. Pseudowires are then signaled to demultiplex traffic between VPLS routing instances. These PE routers exchange the pseudowire label, the MPLS label that acts as the VPLS pseudowire demultiplexer field, by using LDP forwarding equivalence classes (FECs). Tunnels based on both MPLS and generic routing encapsulation (GRE) are supported.

NOTE: You cannot configure both BGP signaling and LDP signaling for the same VPLS routing instance. If you attempt to configure the statements that enable BGP signaling for the VPLS routing instance (the site, site-identifier, and site-range statements), and the statements that enable LDP signaling for the same instance, neighbor and vpls-id, the commit operation fails.

To enable LDP signaling for the VPLS routing instance, complete the steps in the following sections:

- Configuring LDP Signaling for the VPLS Routing Instance on page 479
- Configuring LDP Signaling on the Router on page 479

Configuring LDP Signaling for the VPLS Routing Instance

To configure the VPLS routing instance to use LDP signaling, you must configure the same VPLS identifier on each PE router participating in the instance. Specify the VPLS identifier with the **vpls-id** statement:

vpls-id *vpls-id*;

You can include this statement at the following hierarchy levels:

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

To configure the VPLS routing instance to use LDP signaling, you also must include the **neighbor** statement to specify each of the neighboring PE routers that are a part of this VPLS domain:

neighbor *neighbor-id*;

You can include this statement at the following hierarchy levels:

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

Configuring LDP Signaling on the Router

To enable LDP signaling, you need to configure LDP on each PE router participating in the VPLS routing instance. A minimal configuration is to enable LDP on the loopback interface, which includes the router identifier (**router-id**), on the PE router using the **interface** statement:

interface *interface-name*;

You can include this statement at the following hierarchy levels:

- **[edit protocols ldp]**
- **[edit logical-systems** *logical-system-name* **protocols ldp]**

You can enable LDP on all the interfaces on the router using the **all** option for the **interfaces** statement. For more information about how to configure LDP, see the *Junos OS MPLS Applications Configuration Guide*.

Configuring VPLS Routing Instance and VPLS Interface Connectivity

You can configure the VPLS routing instance to take down or maintain its VPLS connections depending on the status of the interfaces configured for the VPLS routing instance. By default, the VPLS connection is taken down whenever a customer-facing interface configured for the VPLS routing instance fails. This behavior can be explicitly configured by specifying the **ce** option for the **connectivity-type** statement:

connectivity-type ce;

You can alternatively specify that the VPLS connection remain up so long as an Integrated Routing and Bridging (IRB) interface is configured for the VPLS routing instance by specifying the **irb** option for the **connectivity-type** statement:

connectivity-type irb;

To ensure that the VPLS connection remain up until explicitly taken down, specify the **permanent** option for the **connectivity-type** statement:

connectivity-type permanent;

This option is reserved for use in configuring Layer 2 Wholesale subscriber networks. See the *Broadband Subscriber Management Solutions Guide* for details about configuring a Layer 2 Wholesale network.

You can include this statement at the following hierarchy levels:

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

Configuring the VPLS MAC Table Timeout Interval

You can modify the timeout interval for the VPLS table. We recommend you that configure longer values for small, stable VPLS networks and shorter values for large, dynamic VPLS networks. If the VPLS table does not receive any updates during the timeout interval, the router waits one additional interval before automatically clearing the MAC address entries from the VPLS table.

To modify the timeout interval for the VPLS table, include the **mac-table-aging-time** statement:

mac-table-aging-time *seconds*;

You can include this statement at the following hierarchy levels:

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

NOTE: The mac-table-aging-time statement is not available on MX Series routers.

Configuring the Size of the VPLS MAC Address Table

You can modify the size of the VPLS media access control (MAC) address table. The default table size is 512 MAC addresses, the minimum is 16 addresses, and the maximum is 65,536 addresses.

If the MAC table limit is reached, new MAC addresses can no longer be added to the table. Eventually the oldest MAC addresses are removed from the MAC address table automatically. This frees space in the table, allowing new entries to be added. However, as long as the table is full, new MAC addresses are dropped.

To change the VPLS MAC table size for each VPLS or VPN routing instance, include the **mac-table-size** statement:

mac-table-size *size*;

You can include this statement at the following hierarchy levels:

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

When you include the **mac-table-size** statement, the affected interfaces include all interfaces within the VPLS routing instance, including the local interfaces, the LSI interfaces, and the VT interfaces.

Limiting the Number of MAC Addresses Learned from an Interface

You can configure a limit on the number of MAC addresses learned by a VPLS routing instance using the **mac-table-size** statement. If the MAC table limit is reached, new MAC addresses can no longer be added to the table. Eventually the oldest MAC addresses are removed from the MAC address table automatically. This frees space in the table, allowing new entries to be added. However, as long as the table is full, new MAC addresses are dropped.

Because this limit applies to each VPLS routing instance, the MAC addresses of a single interface can consume all the available space in the table, preventing the routing instance from acquiring addresses from other interfaces.

You can limit the number of MAC addresses learned from each interface configured for a VPLS routing instance. To do so, include the **interface-mac-limit** statement:

interface-mac-limit *limit*;

You can include this statement at the following hierarchy levels:

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

The **interface-mac-limit** statement affects the local interfaces only (the interfaces facing CE devices).

Configuring the **interface-mac-limit** statement at the **[edit routing-instances *routing-instance-name* protocols vpls]** hierarchy level causes the same limit to be applied to all of the interfaces configured for that specific routing instance.

You can also limit the number of MAC addresses learned by a specific interface configured for a VPLS routing instance. This gives you the ability to limit particular interfaces that you expect might generate a lot of MAC addresses.

To limit the number of MAC addresses learned by a specific interface, include the **interface-mac-limit** statement at the following hierarchy levels:

- **[edit routing-instances *routing-instance-name* protocols vpls site *site-name* interfaces *interface-name*]**
- **[edit logical-systems *logical-system-name* routing-instances *routing-instance-name* protocols vpls site *site-name* interfaces *interface-name*]**

The MAC limit configured for an individual interface at this hierarchy level overrides any value configured at the **[edit routing-instances *routing-instance-name* protocols vpls]** hierarchy level. Also, the MAC limit configured using the **mac-table-size** statement can override the limit configured using the **interface-mac-limit** statement.

The MAC address limit applies to customer-facing interfaces only.

Removing Addresses from the MAC Address Database

You can enable MAC flush processing for the VPLS routing instance or for the mesh group under a VPLS routing instance. MAC flush processing removes MAC addresses from the MAC address database that have been learned dynamically. With the dynamically learned MAC addresses removed, MAC address convergence requires less time to complete.

You can clear dynamically learned MAC addresses from the MAC address database by including the **mac-flush** statement:

mac-flush [*explicit-mac-flush-message-options*];

To clear dynamically learned MAC addresses globally across all devices participating in the routing instance, you can include the statement at the following hierarchy levels:

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

To clear the MAC addresses on the routers in a specific mesh group, you can include the statement at the following hierarchy levels:

- **[edit logical-systems *logical-system-name* routing-instances *routing-instance-name* protocols vpls mesh-group *mesh-group-name*]**
- **[edit routing-instances *routing-instance-name* protocols vpls mesh-group *mesh-group-name*]**

For certain cases where MAC flush processing is not initiated by default, you can also specify ***explicit-mac-flush-message-options*** to additionally configure the router to send

explicit MAC flush messages under specific conditions. For a list of the explicit MAC flush message options you can include with this statement, see the summary section for this statement.

Configuring Static Pseudowires for VPLS

You can configure a VPLS domain using static pseudowires. A VPLS domain consists of a set of PE routers that act as a single virtual Ethernet bridge for the customer sites connected to these routers. By configuring static pseudowires for the VPLS domain, you do not need to configure the LDP or BGP protocols that would normally be used for signaling. However, if you configure static pseudowires, any changes to the VPLS network topology have to be managed manually.

Static pseudowires require that you configure a set of in and out labels for each pseudowire configured for the VPLS domain. You still need to configure a VPLS identifier and neighbor identifiers for a static VPLS domain. You can configure both static and dynamic neighbors within the same VPLS routing instance.

To configure a static pseudowire for a VPLS neighbor, include the **static** statement:

```
static {
    incoming-label label;
    outgoing-label label;
}
```

You must configure an incoming and outgoing label for the static pseudowire using the **incoming-label** and **outgoing-label** statements. These statements identify the static pseudowire's incoming traffic and destination.

To configure a static pseudowire for a VPLS neighbor, include the **static** statement at the **[edit routing-instances** *routing-instance-name* **protocols vpls neighbor** *address***]** hierarchy level.

You can also configure the **static** statement for a backup neighbor (if you configure the neighbor as static the backup must also be static) by including it at the **[edit routing-instances** *routing-instance-name* **protocols vpls neighbor** *address* **backup-neighbor** *address***]** hierarchy level and for a mesh group by including it at the **[edit routing-instances** *routing-instance-name* **protocols vpls mesh-group** *mesh-group-name* **neighbor** *address***]** hierarchy level.

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

To enable static VPLS on a router, you need to either configure a virtual tunnel interface (requires the router to have a tunnel services PIC) or you can configure a label switching interface (LSI). To configure an LSI, include the **no-tunnel-services** statement at the **[edit protocols vpls static-vpls]** hierarchy level. For more information, see "Configuring VPLS Without a Tunnel Services PIC" on page 492.

NOTE: Static pseudowires for VPLS using an LSI is supported on MX series routers only. For M series and T series routers, a tunnel services PIC is required.

If you issue a **show vpls connections** command, static neighbors are displayed with **"SN"** next to their addresses in the command output.

Related Documentation

- Configuring VPLS Without a Tunnel Services PIC on page 492

Configuring EXP-Based Traffic Classification for VPLS

You can enable EXP classification on traffic entering core facing VPLS LSI interfaces on a VPLS routing instance by configuring either a logical tunnel interface (**lt-**) or the **no-tunnel-services** statement. By configuring either of these, a default EXP classifier is enabled on every core facing interface that includes **family mpls** in its configuration. This feature works on MX Series routers only. You can configure an EXP classifier explicitly at the **[edit class-of-service]** hierarchy level. For more information about EXP classifiers, see the *Junos OS Class of Service Configuration Guide*.

To enable EXP classification on traffic entering core facing VPLS LSI interfaces on a VPLS routing instance, include the **no-tunnel-services** statement:

no-tunnel-services;

You can include this statement at the following hierarchy levels:

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

Configuring Interfaces for VPLS Routing

On each PE router and for each VPLS routing instance, specify which interfaces are intended for the VPLS traffic traveling between PE and CE routers. To specify the interface for VPLS traffic, include the **interface** statement in the routing instance configuration:

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***]**

You must also define each interface by including the following statements:

```
vlan-tagging;
encapsulation encapsulation-type;
unit logical-unit-number {
   vlan-id vlan-id-number;
   family vpls;
}
```

You can include these statements at the following hierarchy levels:

- **[edit interfaces** *interface-name***]**

- **[edit logical-systems** *logical-system-name* **interfaces** *interface-name***]**

The following sections provide enough information to enable you to configure interfaces for VPLS routing. For detailed information about configuring interfaces and the statements at the **[edit interfaces]** hierarchy level, see the *Junos OS Network Interfaces Configuration Guide*.

To configure an interface for VPLS, you perform the steps in the following sections:

Configuring the Interface Name

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

physical.*logical*

For example, in **ge-1/2/1.2**, **ge-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.

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 you configure a specific interface for VPLS routing at the **[edit routing-instances** *routing-instance-name***]** hierarchy level, the latter interface statement takes precedence and the interface is used exclusively for VPLS.

If you explicitly configure the same interface name at both the **[edit protocols]** and **[edit routing-instances** *routing-instance-name***]** hierarchy levels and then attempt to commit the configuration, the commit operation fails.

Configuring the VPLS Interface Encapsulation

You need to specify an encapsulation type for each PE-router-to-CE-router interface configured for VPLS. This section describes the **encapsulation** statement configuration options available for VPLS. For a full description of all of the options available for this statement, see the *Junos OS Network Interfaces Configuration Guide*.

To configure the encapsulation type on the physical interface, include the **encapsulation** statement:

encapsulation (ethernet-vpls | extended-vlan-vpls | vlan-vpls);

You can include the **encapsulation** statement for physical interfaces at the following hierarchy levels:

- **[edit interfaces *interface-name*]**
- **[edit logical-systems *logical-system-name* interfaces *interface-name*]**

You can configure the following physical interface encapsulations for VPLS routing instances:

- **ethernet-vpls**—Use Ethernet VPLS encapsulation on Ethernet interfaces that have VPLS enabled and that must accept packets carrying standard Tag Protocol ID (TPID) values. On M Series routers (except the M320 router), the 4-port Fast Ethernet TX PIC and the 1-port, 2-port, and 4-port, 4-slot Gigabit Ethernet PICs can use the Ethernet VPLS encapsulation type.
- **extended-vlan-vpls**—Use extended virtual LAN (VLAN) VPLS encapsulation on Ethernet interfaces that have VLAN 802.1Q tagging and VPLS enabled and that must accept packets carrying TPIDs 0x8100, 0x9100, and 0x9901. On M Series routers (except the M320 router), the 4-port Fast Ethernet TX PIC and the 1-port, 2-port, and 4-port, 4-slot Gigabit Ethernet PICs can use the Ethernet VPLS encapsulation type.

  NOTE: The built-in Gigabit Ethernet PIC on an M7i router does not support extended VLAN VPLS encapsulation.

- **vlan-vpls**—Use VLAN VPLS encapsulation on Ethernet interfaces with VLAN tagging and VPLS enabled. Interfaces with VLAN VPLS encapsulation accept packets carrying standard TPID values only. On M Series routers (except the M320 router), the 4-port Fast Ethernet TX PIC and the 1-port, 2-port, and 4-port, 4-slot Gigabit Ethernet PICs can use the Ethernet VPLS encapsulation type.

To configure the encapsulation type for logical interfaces, include the **encapsulation** statement:

```
encapsulation (ether-vpls-over-atm-llc | vlan-vpls);
```

You can include the **encapsulation** statement for logical interfaces at the following hierarchy levels:

- **[edit interfaces *interface-name* unit *number*]**
- **[edit logical-systems logical-system-name interfaces *interface-name* unit *number*]**

You can configure the following logical interface encapsulations for VPLS routing instances:

- **ether-vpls-over-atm-llc**—Use Ethernet VPLS over Asynchronous Transfer Mode (ATM) logical link control (LLC) encapsulation to bridge Ethernet interfaces and ATM interfaces over a VPLS routing instance (as described in RFC 2684, *Multiprotocol Encapsulation over ATM Adaptation Layer 5*). Packets from the ATM interfaces are converted to standard ENET2/802.3-encapsulated Ethernet frames with the frame check sequence

(FCS) field removed. This encapsulation type is supported on ATM intelligent queuing (IQ) interfaces only.

- **vlan-vpls**—Use VLAN VPLS encapsulation on Ethernet interfaces with VLAN tagging and VPLS enabled. Interfaces with VLAN VPLS encapsulation accept packets carrying standard TPID values only. On M Series routers (except the M320 router), the 4-port Fast Ethernet TX PIC and the 1-port, 2-port, and 4-port, 4-slot Gigabit Ethernet PICs can use the Ethernet VPLS encapsulation type.

When you configure the physical interface encapsulation as **vlan-vpls**, you also need to configure the same interface encapsulation for the logical interface. You need to configure the **vlan-vpls** encapsulation on the logical interface because the **vlan-vpls** encapsulation allows you to configure a mixed mode, where some of the logical interfaces use regular Ethernet encapsulation (the default for logical interfaces) and some use **vlan-vpls**. For more information, see the *Junos OS Network Interfaces Configuration Guide*.

Enabling VLAN Tagging

The Junos OS supports receiving and forwarding routed Ethernet frames with 802.1Q virtual local area network (VLAN) tags and running Virtual Router Redundancy Protocol (VRRP) over 802.1Q-tagged interfaces. For VPLS to function properly, configure the router to receive and forward frames with 802.1Q VLAN tags by including the **vlan-tagging** statement at the **[edit interfaces *interface-name*]** hierarchy level:

```
[edit interfaces interface-name]
vlan-tagging;
```

Gigabit Ethernet interfaces can be partitioned; you can assign up to 4095 different logical interfaces, one for each VLAN, but you are limited to a maximum of 1024 VLANs on any single Gigabit Ethernet or 10-Gigabit Ethernet port. Fast Ethernet interfaces can also be partitioned, with a maximum of 1024 logical interfaces for the 4-port FE PIC and 16 logical interfaces for the M40e router. Table 10 on page 487 lists VLAN ID range by interface type.

Table 10: VLAN ID Range by Interface Type

Interface Type	VLAN ID Range
Fast Ethernet	512 through 1023
Gigabit Ethernet	512 through 4094

Configuring VLAN IDs for Logical Interfaces

You can bind a VLAN identifier to a logical interface by including the **vlan-id** statement:

```
vlan-id number;
```

You can include this statement at the following hierarchy levels:

- **[edit interfaces *interface-name* unit *logical-unit-number*]**
- **[edit logical-systems *logical-system-name* interfaces *interface-name* unit *logical-unit-number*]**

You can also configure a logical interface to forward packets and learn MAC addresses within each VPLS routing instance configured with a VLAN ID that matches a VLAN ID specified in a list using the **vlan-id-list** statement. VLAN IDs can be entered individually using a space to separate each ID, entered as an inclusive list separating the starting VLAN ID and ending VLAN ID with a hyphen, or a combination of both.

For example, to configure the VLAN IDs 20 and 45 and the range of VLAN IDs between 30 and 40, issue the following command from the CLI:

set interfaces ge-1/0/1 unit 1 vlan-id-list [20 30-40 45];

To configure a list of VLAN IDs for a logical interface, include the **vlan-id-list** statement:

vlan-id-list *list-of-vlan-ids*;

You can include this statement at the following hierarchy levels:

- **[edit interfaces** *interface-name* **unit** *logical-unit-number***]**
- **[edit logical-systems** *logical-system-name* **interfaces** *interface-name* **unit** *logical-unit-number***]**

For more information about how to configure VLANs, see the *Junos OS Network Interfaces Configuration Guide*. For detailed information about how VLAN identifiers in a VPLS routing instance are processed and translated, see the *MX Series Layer 2 Configuration Guide*.

Configuring Aggregated Ethernet Interfaces for VPLS

You can configure aggregated Ethernet interfaces between CE devices and PE routers for VPLS routing instances. Traffic is load-balanced across all of the links in the aggregated interface. If one or more links in the aggregated interface fails, the traffic is switched to the remaining links.

For more information about how aggregated Ethernet interfaces function in the context of VPLS, see “VPLS and Aggregated Ethernet Interfaces” on page 460.

To configure aggregated Ethernet interfaces for VPLS, configure the interface for the VPLS routing instance as follows:

```
interfaces aex {
    vlan-tagging;
    encapsulation encapsulation-type;
    unit logical-unit-number {
        vlan-id number;
    }
}
```

You can configure the following physical link-layer encapsulation types for the VPLS aggregated Ethernet interface:

- **ethernet-vpls**
- **extended-vlan-vpls**
- **flexible-ethernet-services**
- **vlan-vpls**

For the **interface** configuration statement, in **ae*x***, the ***x*** represents the interface instance number to complete the link association; ***x*** can be from 0 through 127, for a total of 128 aggregated interfaces.

For more information about how to configure aggregated Ethernet interfaces, see the *Junos OS Network Interfaces Configuration Guide*.

The aggregated Ethernet interface must also be configured for the VPLS routing instance as shown in the following example:

```
[edit]
routing-instances {
   green {
      instance-type vpls;
      interface ae0.0;
      route-distinguisher 10.255.234.34:1;
      vrf-target target:11111:1;
      protocols {
         vpls {
            site-range 10;
            site green3 {
               site-identifier 3;
            }
         }
      }
   }
}
```

Interface **ae0.0** represents the aggregated Ethernet interface in the routing instance configuration. The VPLS routing instance configuration is otherwise standard.

Configuring VPLS Load Balancing

By default, when there are multiple equal-cost paths to the same destination for the active route, the Junos OS 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, the next-hop address is reselected using the hash algorithm.

You can configure the Junos OS so that, for the active route, all next-hop addresses for a destination are installed in the forwarding table. This feature is called per-packet load balancing. You can use load balancing to spread traffic across multiple paths between routers. You can also configure per-packet load balancing to optimize VPLS traffic flows across multiple paths.

You can load-balance VPLS traffic based on Layer 2 media access control (MAC) information, IP information and MPLS labels, or MPLS labels only.

NOTE: This feature is not supported on J Series Services Routers or MX Series routers. VPLS load balancing based on IP information and MPLS labels is supported only on the M120 and M320 routers.

To optimize VPLS traffic flows across multiple paths, include the **family multiservice** statement at the **[edit forwarding-options hash-key]** hierarchy level:

```
family multiservice {
  destination-mac;
  label-1;
  label-2;
  payload {
    ip {
      layer-3-only;
    }
  }
  source-mac;
}
```

To load-balance based on Layer 2 information, include the following configuration options:

- **destination-mac**—Include the destination MAC address in the hash key used to load-balance the VPLS traffic.
- **source-mac**—Include the source MAC address in the hash key used to load-balance the VPLS traffic.

You can include the source MAC address in the hash key, the destination MAC address, or both.

For IPv4 traffic, only the IP source and destination addresses are included in the hash key. For MPLS and IPv4 traffic, one or two MPLS labels and IPv4 source and destination addresses are included. For MPLS Ethernet pseudowires, only one or two MPLS labels are included in the hash key. Optionally, you can include only Layer 3 information the IPv4 payload in the hash key.

To load-balance based on IP information and MPLS labels, include the following configuration options:

- **label-1**—Include the first MPLS label in the hash key used to load-balance VPLS traffic.
- **label-2**—Include the second MPLS label in the hash key used to load-balance VPLS traffic.
- **payload**—Include bits from the IP payload in the hash key used to load-balance VPLS traffic.
- **ip**—Include the IP address of the IPv4 payload in the hash key used to load-balance VPLS traffic.
- **layer-3-only**—Include only Layer 3 information in the hash key used to load-balance VPLS traffic

For more information about how to configure per-packet load balancing, see the *Junos OS Policy Framework Configuration Guide*.

Configuring VPLS Fast Reroute Priority

When a path is rerouted after a link failure by using the MPLS fast reroute feature, the router repairs the affected next hops by switching them from the active label switched path (LSP) to the standby LSP. To specify the order in which the router repairs next hops and restores traffic convergence for VPLS routing instances after a fast reroute event, you can use the **fast-reroute-priority** statement to configure **high**, **medium**, or **low** fast reroute priority for a VPLS routing instance. By default, the fast reroute priority for a VPLS routing instance is **low**.

The router repairs next hops and restores known unicast, unknown unicast, broadcast, and multicast traffic for VPLS routing instances in the following order, based on the fast reroute priority configuration:

1. The router repairs next hops for high-priority VPLS routing instances.
2. The router repairs next hops for medium-priority VPLS routing instances.
3. The router repairs next hops for low-priority VPLS routing instances.

Because the router repairs next hops for VPLS routing instances configured with **high** fast reroute priority first, the traffic traversing high-priority VPLS instances is restored faster than the traffic for VPLS instances configured with **medium** or **low** fast reroute priority. The ability to prioritize specific VPLS routing instances for faster convergence and traffic restoration enables service providers to offer differentiated service levels to their customers.

Within a particular fast reroute priority level (**high**, **medium**, or **low**), the router follows no particular order for traffic restoration of VPLS routing instances.

NOTE: VPLS fast reroute priority is not supported on EX Series switches or J Series routers.

To configure **high**, **medium**, or **low** fast reroute priority for a VPLS routing instance, include the **fast-reroute-priority** statement:

fast-reroute-priority (high | medium | low);

You can include this statement at the following hierarchy levels:

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

You can configure fast reroute priority only for routing instances with the **instance-type** set to **vpls**. If you attempt to configure fast reroute priority for a routing instance with an **instance-type** other than **vpls**, the router displays a warning message and the configuration fails.

The following example snippet shows configuration of **high** fast reroute priority for a VPLS routing instance named **test-vpls**:

```
test-vpls {
    instance-type vpls;
    forwarding-options {
        fast-reroute-priority high;
    }
}
```

To display the fast reroute priority setting configured for a VPLS routing instance, use the **show route instance detail** operational command. For information about using this command, see the *Junos OS Routing Protocols and Policies Command Reference*.

Configuring VPLS Without a Tunnel Services PIC

VPLS normally uses a dynamic virtual tunnel logical interface on a Tunnel Services PIC to model traffic from a remote site (a site on a remote PE router that is in a VPLS domain). All traffic coming from a remote site is treated as coming in over the virtual port representing this remote site, for the purposes of Ethernet flooding, forwarding, and learning. An MPLS lookup based on the inner VPN label is done on a PE router. The label is stripped and the Layer 2 Ethernet frame contained within is forwarded to a Tunnel Services PIC. The PIC loops back the packet and then a lookup based on Ethernet MAC addresses is completed. This approach requires that the router have a Tunnel Services PIC and that the PE router complete two protocol lookups.

You can configure VPLS without a Tunnel Services PIC by configuring the **no-tunnel-services** statement. This statement creates a label-switched interface (LSI) to provide VPLS functionality. An LSI MPLS label is used as the inner label for VPLS. This label maps to a VPLS routing instance. On the PE router, the LSI label is stripped and then mapped to a logical LSI interface. The Layer 2 Ethernet frame is then forwarded using the LSI interface to the correct VPLS routing instance.

By default, VPLS requires a Tunnel Services PIC. To configure VPLS on a router without a Tunnel Services PIC and create an LSI, include the **no-tunnel-services** statement:

```
no-tunnel-services;
```

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

To configure a VPLS routing instance on a router without a tunnel services PIC, include the **no-tunnel-services** statement at the **[edit routing-instances *routing-instance-name* protocols vpls]** hierarchy level. To configure static VPLS on a router without a tunnel services PIC, include the **no-tunnel-services** statement at the **[edit protocols vpls static-vpls]** hierarchy level.

When you configure VPLS without a Tunnel Services PIC by including the **no-tunnel-services** statement, the following limitations apply:

- An Enhanced FPC is required.
- ATM1 interfaces are not supported.

- Aggregated SONET/SDH interfaces are not supported as core-facing interfaces.
- Channelized interfaces are not supported as core-facing interfaces.
- GRE-encapsulated interfaces are not supported as core-facing interfaces.

Related Documentation

- Configuring Static Pseudowires for VPLS on page 483

Configuring an Ethernet Switch as the CE Device

For VPLS configurations, the CE device does not necessarily need to be a router. You can link the PE routers directly to Ethernet switches. However, there are a few configuration issues to be aware of:

- When you configure VPLS routing instances and establish two or more connections between a CE Ethernet switch and a PE router, you must enable the Spanning Tree Protocol (STP) on the switch to prevent loops.
- The Junos OS allows standard Bridge Protocol Data Unit (BPDU) frames to pass through emulated Layer 2 connections, such as those configured with Layer 2 VPNs, Layer 2 circuits, and VPLS instances. However, CE Ethernet switches that generate proprietary BPDU frames might not be able to run STP across Juniper Networks routing platforms configured for these emulated Layer 2 connections.

Mapping VPLS Traffic to Specific LSPs

You can map VPLS traffic to specific LSPs by configuring forwarding table policies. This procedure is optional but can be useful. The following example illustrates how you can map lower priority VPLS routing instances to slower LSPs while mapping other higher priority VPLS routing instances to faster LSPs. In this example configuration, **a-to-b1** and **a-to-c1** are high-priority LSPs between the PE routers, while **a-to-b2** and **a-to-c2** are low-priority LSPs between the PE routers.

To map VPLS traffic, include the **policy-statement vpls-priority** statement:

```
policy-statement vpls-priority {
  term a {
    from {
      rib mpls.0;
      community company-1;
    }
    then {
      install-nexthop lsp [ a-to-b1 a-to-c1 ];
      accept;
    }
  }
  term b {
    from {
      rib mpls.0;
      community company-2;
    }
    then {
```

```
        install-nexthop lsp-regex [ "^a-to-b2$" "^a-to-c2$" ];
        accept;
      }
    }
  }
  community company-1 members target:11111:1;
  community company-2 members target:11111:2;
```

You can include the **policy-statement vpls-priority** statement at the following hierarchy levels:

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

Include the **export** statement to apply the **vpls-priority** policy to the forwarding table:

```
export vpls-priority;
```

You can include this statement at the following hierarchy levels:

- **[edit routing-options forwarding-table]**
- **[edit logical-systems *logical-system-name* routing-options forwarding-table]**

For more information about how to configure routing policies, see the *Junos OS Policy Framework Configuration Guide*.

Configuring Firewall Filters and Policers for VPLS

You can configure both firewall filters and policers for VPLS. Firewall filters allow you to filter packets based on their components and to perform an action on packets that match the filter. Policers allow you to limit the amount of traffic that passes into or out of an interface.

VPLS filters and policers act on a Layer 2 frame that includes the media access control (MAC) header (after any VLAN rewrite or other rules are applied), but does not include the cyclical redundancy check (CRC) field.

You can apply VPLS filters and policers on the PE router to customer-facing interfaces only.

The following sections explain how to configure filters and policers for VPLS:

- Configuring a VPLS Filter on page 494
- Configuring a VPLS Policer on page 498

Configuring a VPLS Filter

To configure a filter for VPLS, include the **filter** statement at the **[edit firewall family vpls]** hierarchy level:

```
[edit firewall family vpls]
filter filter-name {
  interface-specific;
```

```
    term term-name {
      from {
        match-conditions;
      }
      then {
        actions;
      }
    }
}
```

For more information about how to configure firewall filters, see the *Junos OS Policy Framework Configuration Guide*. For information on how to configure a VPLS filter match condition, see "Configuring VPLS Match Conditions" on page 498.

To configure a filter for VPLS traffic, complete the following tasks:

Configuring an Interface-Specific Counter for VPLS

When you configure a firewall filter for VPLS and apply it to multiple interfaces, you can specify individual counters specific to each interface. This allows you to collect separate statistics on the traffic transiting each interface.

To generate an interface-specific counter for VPLS, you configure the **interface-specific** statement. A separate instantiation of the filter is generated. This filter instance has a different name (based on the interface name) and collects statistics on the interface specified only.

To configure interface-specific counters, include the **interface-specific** statement at the **[edit firewall family vpls filter *filter-name*]** hierarchy level:

```
[edit firewall family vpls filter filter-name]
interface-specific;
```

NOTE: The counter name is restricted to 24 bytes. If the renamed counter exceeds this maximum length, it might be rejected.

For more information about the **interface-specific** statement and an example of how to configure it, see the *Junos OS Policy Framework Configuration Guide*.

Configuring an Action for the VPLS Filter

You can configure the following actions for a VPLS filter at the **[edit firewall family vpls filter *filter-name* term *term-name* then]** hierarchy level: **accept**, **count**, **discard**, **forwarding-class**, **loss-priority**, **next**, **policer**.

Configuring VPLS FTFs

Forwarding table filters (FTFs) are filters configured for forwarding tables. For VPLS, they are attached to the destination MAC (DMAC) forwarding table of the VPLS routing instance. You define VPLS FTFs in the same manner as any other type of FTF. You can only apply a VPLS FTF as an input filter.

To specify a VPLS FTF, include the **filter input** statement at the **[edit routing-instance *routing-instance-name* forwarding-options family vpls]** hierarchy level:

```
[edit routing-instance routing-instance-name forwarding-options family vpls]
filter input filter-name;
```

For the statement summaries of these statements, see the *Junos OS Policy Framework Configuration Guide*.

Changing Precedence for Spanning-Tree BPDU Packets

Spanning tree BPDU packets are automatically set to a high precedence. The queue number on these packets is set to 3. On M Series routers (except the M320 router) by default, a queue value of 3 indicates high precedence. To enable this higher precedence on BPDU packets, an instance-specific BPDU precedence filter named **default_bpdu_filter** is automatically attached to the VPLS DMAC table. This filter places a high precedence on all packets sent to **01:80:c2:00:00:00/24**.

You can overwrite this filter by configuring a VPLS FTF filter and applying it to the VPLS routing instance. For more information, see “Configuring VPLS FTFs” on page 496 and “Applying a VPLS Filter to a VPLS Routing Instance” on page 497.

Applying a VPLS Filter to an Interface

To apply a VPLS filter to an interface, include the **filter** statement:

```
filter {
    input input-filter-name;
    output output-filter-name;
    group index;
}
```

You can include this statement at the following hierarchy levels:

- **[edit interfaces *interface-name* unit *number* family vpls]**
- **[edit logical-systems *logical-system-name* interfaces *interface-name* unit *number* family vpls]**

In the **input** statement, list the name of the VPLS filter to be evaluated when packets are received on the interface. In the **output** statement, list the name of the VPLS filter to be evaluated when packets are transmitted on the interface.

NOTE: For output interface filters, MAC addresses are learned after the filter action is completed. When an output interface filter's action is **discard**, the packet is dropped before the MAC address is learned. However, an input interface filter learns the MAC address before discarding the packet.

For the statement summaries for these statements, see the *Junos OS Network Interfaces Configuration Guide*.

Applying a VPLS Filter to a VPLS Routing Instance

You can apply a VPLS filter to a VPLS routing instance. The filter checks traffic passing through the specified routing instance.

Input routing instance filters learn the MAC address before the filter action is completed, so if the filter action is **discard**, the MAC address is learned before the packet is dropped.

To apply a VPLS filter to packets arriving at a VPLS routing instance and specify the filter, include the **filter input** statement at the **[edit routing-instances *routing-instance-name* forwarding-options family vpls]** hierarchy level:

```
[edit routing-instances routing-instance-name forwarding-options family vpls]
filter input input-filter-name;
```

Configuring a Filter for Flooded Traffic

You can configure a VPLS filter to filter flooded packets. CE routers typically flood the following types of packets to PE routers in VPLS routing instances:

- Layer 2 broadcast packets
- Layer 2 multicast packets
- Layer 2 unicast packets with an unknown destination MAC address
- Layer 2 packets with a MAC entry in the DMAC routing table

You can configure filters to manage how these flooded packets are distributed to the other PE routers in the VPLS routing instance.

To apply a flooding filter to packets arriving at the PE router in the VPLS routing instance, and specify the filter, include the **flood input** statement:

```
flood input filter-name;
```

You can include this statement at the following hierarchy levels:

- **[edit routing-instances *routing-instance-name* forwarding-options family vpls]**
- **[edit logical-systems *logical-system-name* routing-instances *routing-instance-name* forwarding-options family vpls]**

Configuring a VPLS Policer

You can configure a policer for VPLS traffic. The VPLS policer configuration is similar to the configuration of any other type of policer.

VPLS policers have the following characteristics:

- You cannot police the default VPLS routes stored in the flood table from PE router–sourced flood traffic.
- When specifying policing bandwidth, the VPLS policer considers all Layer 2 bytes in a packet to determine the packet length.

To configure a VPLS policer, include the **policer** statement at the **[edit firewall]** hierarchy level:

```
[edit firewall]
policer policer-name {
    bandwidth-limit limit;
    burst-size-limit limit;
    then action;
}
```

For the statement summaries of these statements and more information about how to configure policers, see the *Junos OS Policy Framework Configuration Guide*.

To apply a VPLS policer to an interface, include the **policer** statement:

```
policer {
    input input-policer-name;
    output output-policer-name;
}
```

You can include this statement at the following hierarchy levels:

- **[edit interfaces** *interface-name* **unit** *number* **family vpls]**
- **[edit logical-systems** *logical-system-name* **interfaces** *interface-name* **unit** *number* **family vpls**

In the **input** statement, list the name of the VPLS policer to be evaluated when packets are received on the interface. In the **output** statement, list the name of the VPLS policer to be evaluated when packets are transmitted on the interface. This type of VPLS policer can only apply to unicast packets. For information about how to filter flood packets, see “Configuring a Filter for Flooded Traffic” on page 497.

For the statement summaries for these statements, see the *Junos OS Network Interfaces Configuration Guide*.

Configuring VPLS Match Conditions

In the **from** statement in the VPLS filter term, you specify conditions that the packet must match for the action in the **then** statement to be taken. All conditions in the **from** statement must match for the action to be taken. The order in which you specify match

conditions is not important, because a packet must match all the conditions in a term for a match to occur.

If you specify no match conditions in a term, that term matches all packets.

An individual condition in a **from** statement can contain a list of values. For example, you can specify numeric ranges or multiple source or destination addresses. When a condition defines a list of values, a match occurs if one of the values in the list matches the packet.

Individual conditions in a **from** statement can be negated. When you negate a condition, you are defining an explicit mismatch. For example, the negated match condition for **forwarding-class** is **forwarding-class-except**. If a packet matches a negated condition, it is immediately considered not to match the **from** statement, and the next term in the filter is evaluated, if there is one; if there are no more terms, the packet is discarded.

Not all match conditions for VPLS traffic are supported on all routing platforms. A number of match conditions for VPLS traffic are supported only on MX Series Ethernet Services Routers, as noted in the Table 11 on page 499.

To specify the match conditions for a VPLS filter term, include the ***match-conditions*** statement at the **[edit firewall family vpls filter *filter-name* term *term-name* from]** hierarchy level.

Table 11 on page 499 describes the firewall filter match conditions supported for VPLS.

Table 11: VPLS Firewall Filter Match Conditions

Match Condition	Description
destination mac-address *address*	Destination media access control (MAC) address of a VPLS packet.
destination-port *number*	(MX Series routers only) TCP or UDP destination port field. You cannot specify both the **port** and **destination-port** match conditions in the same term.
destination-port-except *number*	(MX Series routers only) Do not match on the TCP or UDP destination port field. You cannot specify both the **port** and **destination-port** match conditions in the same term.
destination-prefix-list *name*	(MX Series routers only) Destination prefixes in the specified list name. Specify the name of a prefix list defined at the **[edit policy-options prefix-list *prefix-list-name*]** hierarchy level. NOTE: VPLS prefix lists support only IPV4 addresses. IPV6 addresses included in a VPLS prefix list will be discarded.

Table 11: VPLS Firewall Filter Match Conditions *(continued)*

Match Condition	Description
dscp *number*	(MX Series routers only) Differentiated Services code point (DSCP). The DiffServ protocol uses the type-of-service (ToS) byte in the IP header. The most significant 6 bits of this byte form the DSCP. For more information, see the *Junos OS Class of Service Configuration Guide*. You can specify a numeric value from **0** through **63**. To specify the value in hexadecimal form, include **0x** as a prefix. To specify the value in binary form, include **b** as a prefix. In place of the numeric value, you can specify one of the following text synonyms (the field values are also listed): • RFC 3246, *An Expedited Forwarding PHB (Per-Hop Behavior)*, defines one code point: **ef** (46). • RFC 2597, *Assured Forwarding PHB Group*, defines 4 classes, with 3 drop precedences in each class, for a total of 12 code points: **af11** (10), **af12** (12), **af13** (14), **af21** (18), **af22** (20), **af23** (22), **af31** (26), **af32** (28), **af33** (30), **af41** (34), **af42** (36), **af43** (38)
dscp-except *number*	(MX Series routers only) Do not match on the DSCP.
ether-type *number*	Ethernet type field of a VPLS packet.
ether-type-except *number*	Do not match on the Ethernet type field of a VPLS packet.
forwarding-class *class*	Forwarding class. Specify **assured-forwarding**, **best-effort**, **expedited-forwarding**, or **network-control**.
forwarding-class-except *class*	Do not match on the forwarding class. Specify **assured-forwarding**, **best-effort**, **expedited-forwarding**, or **network-control**.
icmp-code *number*	(MX Series routers only) ICMP code field. This value or keyword provides more specific information than **icmp-type**. Because the value's meaning depends upon the associated **icmp-type**, you must specify **icmp-type** along with **icmp-code**. For more information, see Overview of Protocol Match Conditions. In place of the numeric value, you can specify one of the following text synonyms (the field values are also listed). The keywords are grouped by the ICMP type with which they are associated: • parameter-problem: **ip-header-bad** (0), **required-option-missing** (1) • redirect: **redirect-for-host** (1), **redirect-for-network** (0), **redirect-for-tos-and-host** (3), **redirect-for-tos-and-net** (2) • time-exceeded: **ttl-eq-zero-during-reassembly** (1), **ttl-eq-zero-during-transit** (0) • unreachable: **communication-prohibited-by-filtering** (13), **destination-host-prohibited** (10), **destination-host-unknown** (7), **destination-network-prohibited** (9), **destination-network-unknown** (6), **fragmentation-needed** (4), **host-precedence-violation** (14), **host-unreachable** (1), **host-unreachable-for-TOS** (12), **network-unreachable** (0), **network-unreachable-for-TOS** (11), **port-unreachable** (3), **precedence-cutoff-in-effect** (15), **protocol-unreachable** (2), **source-host-isolated** (8), **source-route-failed** (5)

Table 11: VPLS Firewall Filter Match Conditions *(continued)*

Match Condition	Description
icmp-code-except *number*	(MX Series routers only) Do not match on the ICMP code field.
icmp-type *number*	(MX Series routers only) ICMP packet type field. Normally, you specify this match in conjunction with the **protocol** match statement to determine which protocol is being used on the port. For more information, see Overview of Protocol Match Conditions. In place of the numeric value, you can specify one of the following text synonyms (the field values are also listed): **echo-reply** (0), **echo-request** (8), **info-reply** (16), **info-request** (15), **mask-request** (17), **mask-reply** (18), **parameter-problem** (12), **redirect** (5), **router-advertisement** (9), **router-solicit** (10), **source-quench** (4), **time-exceeded** (11), **timestamp** (13), **timestamp-reply** (14), or **unreachable** (3).
icmp-type-except *number*	(MX Series routers only) Do not match on the ICMP packet type field.
interface-group *group-name*	Interface group on which the packet was received. An interface group is a set of one or more logical interfaces. For information about configuring interface groups, see Applying Firewall Filters to Interfaces.
interface-group-except *group-name*	Do not match on the interface group.
interface-set *interface-set-name*	(MX Series routers and routers with Enhanced IQ2 [IQ2E] PICs only) Interface set on which the packet was received. An interface set is a set of logical interfaces used to configure hierarchical class-of- service schedulers.
ip-address *address*	(MX Series routers only) 32-bit address that supports the standard syntax for IPv4 addresses.
ip-destination-address *address*	(MX Series routers only) 32-bit address that is the final destination node address for the packet.
ip-precedence *ip-precedence-field*	(MX Series routers only) IP precedence field. In place of the numeric field value, you can specify one of the following text synonyms (the field values are also listed): **critical-ecp** (0xa0), **flash** (0x60), **flash-override** (0x80), **immediate** (0x40), **internet-control** (0xc0), **net-control** (0xe0), **priority** (0x20), or **routine** (0x00).
ip-precedence-except *ip-precedence-field*	(MX Series routers only) Do not match on the IP precedence field.
ip-protocol *number*	(MX Series routers only) IP protocol field.
ip-protocol-except *number*	(MX Series routers only) Do not match on the IP protocol field.
ip-source-address *address*	(MX Series routers only) IP address of the source node sending the packet.
learn-vlan-1p-priority *number*	(MX Series routers only) IEEE 802.1p learned VLAN priority field. Specify a single value or multiple values from **0** through **7**.
learn-vlan-1p-priority-except *number*	(MX Series routers only) Do not match on the IEEE 802.1p learned VLAN priority field. Specify a single value or multiple values from **0** through **7**.

Table 11: VPLS Firewall Filter Match Conditions *(continued)*

Match Condition	Description
learn-vlan-id *number*	(MX Series routers only) VLAN identifier used for MAC learning.
learn-vlan-id-except *number*	(MX Series routers only) Do not match on the VLAN identifier used for MAC learning.
loss-priority *level*	Packet loss priority (PLP) level. Specify a single level or multiple levels: **low**, **medium-low**, **medium-high**, or **high**. Supported on MX Series routers; M120 and M320 routers; and M7i and M10i routers with the Enhanced CFEB (CFEB-E). On M320 routers, you must enable the **tricolor** statement at the **[edit class-of-service]** hierarchy level to commit a PLP configuration with any of the four levels specified. If the **tricolor** statement is not referenced, you can only configure the **high** and **low** levels. This applies to all protocol families. For information about using behavior aggregate (BA) classifiers to set the PLP level of incoming packets, see the *Junos OS Class of Service Configuration Guide*.
loss-priority-except *level*	Do not match on the packet loss priority level. Specify a single level or multiple levels: **low**, **medium-low**, **medium-high**, or **high**. For information about using behavior aggregate (BA) classifiers to set the PLP level of incoming packets, see the *Junos OS Class of Service Configuration Guide*.
port *number*	(MX Series routers only) TCP or UDP source or destination port. You cannot specify both the **port** match condition and either the **destination-port** or **source-port** match condition in the same term.
port-except *number*	(MX Series routers only) Do not match on the TCP or UDP source or destination port. You cannot specify both the **port** match condition and either the **destination-port** or **source-port** match condition in the same term.
prefix-list *name*	(MX Series routers only) Destination or source prefixes in the specified list name. Specify the name of a prefix list defined at the **[edit policy-options prefix-list** *prefix-list-name*] hierarchy level. NOTE: VPLS prefix lists support only IPV4 addresses. IPV6 addresses included in a VPLS prefix list will be discarded.
source-mac-address *address*	Source MAC address of a VPLS packet.
source-port *number*	(MX Series routers only) TCP or UDP source port field. You cannot specify the **port** and **source-port** match conditions in the same term.
source-port-except *number*	(MX Series routers only) Do not match on the TCP or UDP source port field. You cannot specify the **port** and **source-port** match conditions in the same term.
source-prefix-list *name*	(MX Series routers only) Source prefixes in the specified prefix list. Specify a prefix list name defined at the **[edit policy-options prefix-list** *prefix-list-name***]** hierarchy level. NOTE: VPLS prefix lists support only IPV4 addresses. IPV6 addresses included in a VPLS prefix list will be discarded.

Table 11: VPLS Firewall Filter Match Conditions *(continued)*

Match Condition	Description
tcp-flags *flags*	(MX Series routers only) One or more of the following TCP flags: • Bit-name: **fin**, **syn**, **rst**, **push**, **ack**, **urgent** • Numerical value: **0x01** through **0x20** • Text synonym: **tcp-established**, **tcp-initial** You can string together multiple flags using logical operators. Configuring the **tcp-flags** match condition requires that you configure the **next-header-tcp** match condition.
traffic-type *type-name*	(MX Series routers only) Traffic type. Specify **broadcast**, **multicast**, **unknown-unicast**, or **known-unicast**.
traffic-type-except *type-name*	(MX Series routers only) Do not match on the traffic type. Specify **broadcast**, **multicast**, **unknown-unicast**, or **known-unicast**.
user-vlan-1p-priority *number*	IEEE 802.1p user priority field. Specify a single value or multiple values from **0** through **7**.
user-vlan-1p-priority-except *number*	Do not match on the IEEE 802.1p user priority field. Specify a single value or multiple values from **0** through **7**.
user-vlan-id *number*	(MX Series routers only) First VLAN identifier that is part of the payload.
user-vlan-id-except *number*	(MX Series routers only) Do not match on the first VLAN identifier that is part of the payload.
vlan-ether-type *value*	VLAN Ethernet type field of a VPLS packet.
vlan-ether-type-except *value*	Do not match on the VLAN Ethernet type field of a VPLS packet.

Related Documentation
- How to Specify Firewall Filter Match Conditions

Specifying the VT Interfaces Used by VPLS Routing Instances

By default, the Junos OS automatically selects one of the virtual tunnel (VT) interfaces available to the router for de-encapsulating traffic from a remote site. The Junos OS cycles through the currently available VT interfaces, regularly updating the list of available VT interfaces as new remote sites are discovered and new connections are brought up. However, you can also explicitly configure which VT interfaces will receive the VPLS traffic.

By including the **tunnel-services** statement at the **[edit routing-instances** *routing-instance-name* **protocols vpls]** hierarchy level, you can specify that traffic for particular VPLS routing instances be forwarded to specific VT interfaces. Doing so allows you to load-balance VPLS traffic among all the available VT interfaces on the router.

The **tunnel-services** statement includes the following options:

- **devices**—Specifies the VT interfaces acceptable for use by the VPLS routing instance. If you do not configure this option, all VT interfaces available to the router can be used for de-encapsulating traffic for this instance.
- **primary**—Specifies the primary VT interface to be used by the VPLS routing instance. The VT interface specified is used to de-encapsulate all VPLS traffic from the MPLS core network for this routing instance. If the VT interface specified is unavailable, then one of the other acceptable VT interfaces (specified in the devices option) is used for handling the VPLS traffic. If you do not configure this option, any acceptable VT interface can be used to de-encapsulate VPLS traffic from the core.

To specify that traffic for a particular VPLS routing instance be forwarded to specific VT interfaces, include the **tunnel-services** statement:

```
tunnel-services {
  devices device-names;
  primary primary-device-name;
}
```

These statements can be configured at the following hierarchy levels:

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

Configuring VPLS Multihoming

VPLS multihoming allows you to connect a customer site to multiple PE routers to provide redundant connectivity while preventing the formation of Layer 2 loops in the service provider's network. A VPLS site multihomed to two or more PE routers provides redundant connectivity in the event of a PE router-to-CE device link failure or the failure of a PE router. For more information about VPLS multihoming, see "VPLS Multihoming" on page 461.

NOTE: If you want to enable multihoming for a VPLS routing instance, you cannot also enable LDP signaling. You can only enable BGP signaling.

The following sections describe how to configure VPLS multihoming. Some information is also provided on single-homed site configuration versus multihomed site configuration.

VPLS Multihomed Site Configuration

The following describes the requirements for a VPLS multihomed site configuration:

- Assign the same site ID on all PE routers connected to the same CE devices.
- Assign the same route distinguisher on all PE routers connected to the same CE devices.
- Reference all interfaces assigned to the multihomed VPLS site on each PE router. Only one of these interfaces is used to send and receive traffic for this site at a time.
- Either designate a primary interface or allow the router to select the interface to be used as the primary interface.

  If the router selects the interface, the interface used to connect the PE router to the site depends on the order in which interfaces are listed in the PE router's configuration. The first operational interface in the set of configured interfaces is chosen to be the designated interface. If this interface fails, the next interface in the list is selected to send and receive traffic for the site.
- Configure multihoming for the site.

The following configuration shows the statements you need to configure to enable VPLS multihoming:

```
[edit routing-instances routing-instance-name]
instance-type vpls;
interface interface-name;
interface interface-name;
protocols vpls {
   site site-name {
      active-interface {
         any;
         primary interface-name;
      }
      interface interface-name;
      interface interface-name;
      multi-homing;
      site-identifier number;
   }
}
route-distinguisher (as-number:id | ip-address:id);
```

NOTE: If you add a direct connection between CE devices that are multihomed to the same VPLS site on different PE routers, the traffic can loop and a loss of connectivity might occur. We do not recommend this topology.

Most of these statements are explained in more detail in the rest of this chapter. The following sections explain how to configure the statements that are specific to VPLS multihoming:

- Specifying an Interface as the Active Interface on page 506
- Configuring Multihoming on the PE Router on page 506

Specifying an Interface as the Active Interface

You need to specify one of the interfaces for the multihomed site as the primary interface. If there are multiple interfaces, the remaining interfaces are activated only when the primary interface goes down. If no active interfaces are configured at the site level, all traffic for a VPLS site travels through a single, non-multihomed PE router.

You must configure one of the following options for the **active-interface** statement:

- **any**—One configured interface is randomly designated as the active interface for the VPLS site.
- **primary**—Specify the name of the multihomed interface to be used as the primary interface by the VPLS site.

To specify a multihomed interface as the primary interface for the VPLS site, include the **active-interface** statement:

```
active-interface {
  any;
  primary interface-name;
}
```

You can include this statement at the following hierarchy levels:

- **[edit routing-instances *routing-instance-name* protocols vpls site *site-name*]**
- **[edit logical-systems *logical-system-name* routing-instances *routing-instance-name* protocols vpls site *site-name*]**

Configuring Multihoming on the PE Router

When a CE device is connected to the same VPLS site on more than one PE router, include the **multi-homing** statement on all associated PE routers. Configuration of this statement tracks BGP peers. If no BGP peer is available, VPLS deactivates all active interfaces for a site. To specify that the PE router is part of a multihomed VPLS site, include the **multi-homing** statement:

```
multi-homing;
```

You can include this statement at the following hierarchy levels:

- **[edit routing-instances *routing-instance-name* protocols vpls site *site-name*]**
- **[edit logical-systems *logical-system-name* routing-instances *routing-instance-name* protocols vpls site *site-name*]**

Include the **multi-homing** statement on all PE routers associated with a particular VPLS site.

VPLS Single-Homed Site Configuration

All VPLS single-homed sites are connected to the same default VE device. All interfaces in a VPLS routing instance that are not configured as part of a multihomed site are assumed to be single-homed to the default VE device.

Flooding Unknown Traffic Using Point-to-Multipoint LSPs

For a VPLS routing instance, you can flood unknown unicast, broadcast, and multicast traffic using point-to-multipoint (also called *P2MP*) LSPs. By default, VPLS relies upon ingress replication to flood unknown traffic to the members of a VPLS routing instance. This can cause replication of data at routing nodes shared by multiple VPLS members, as shown in Figure 55 on page 507. The flood data is tripled between PE router PE1 and provider router P1 and doubled between provider routers P1 and P2. By configuring point-to-multipoint LSPs to handle flood traffic, the VPLS routing instance can avoid this type of traffic replication in the network, as shown in Figure 56 on page 507.

Figure 55: Flooding Unknown VPLS Traffic Using Ingress Replication

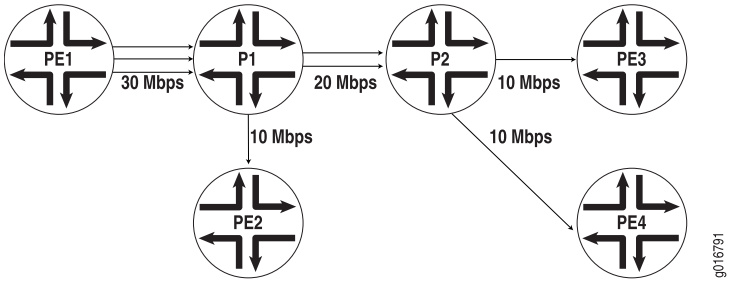
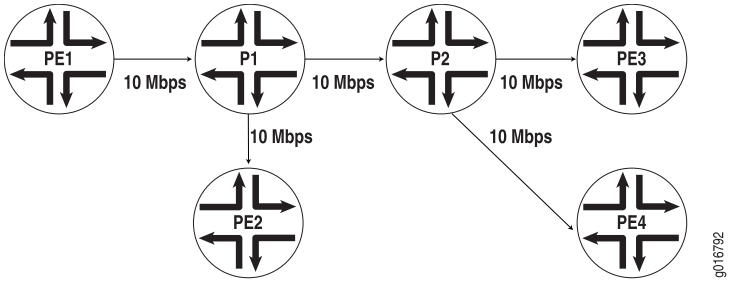


Figure 56: Flooding Unknown VPLS Traffic Using a Point-to-Multipoint LSP



The point-to-multipoint LSP used for VPLS flooding can be either static or dynamic. In either case, for each VPLS routing instance, the PE router creates a dedicated point-to-multipoint LSP. All of the neighbors of the VPLS routing instance are added to the point-to-multipoint LSP when the feature is enabled. If there are *n* PE routers in the VPLS routing instance, *n* point-to-multipoint LSPs are created in the network where each PE router is the root of the point-to-multipoint tree and includes the rest of the *n – 1* PE routers as leaf nodes. If you configured static point-to-multipoint LSPs for flooding, any additional VPLS neighbors added to the routing instance later are not automatically added to the point-to-multipoint LSP. You will need to manually add the new VPLS neighbors to the static point-to-multipoint flooding LSP. If you configure dynamic point-to-multipoint LSPs, whenever VPLS discovers a new neighbor through BGP, a sub-LSP for this neighbor is added to the point-to-multipoint LSP for the routing instance.

This feature can be enabled incrementally on any PE router that is part of a specific VPLS routing instance. The PE routers can then use point-to-multipoint LSPs to flood traffic, whereas other PE routers in the same VPLS routing instance can still use ingress replication

to flood traffic. However, when this feature is enabled on any PE router, you must ensure that all PE routers in the VPLS routing instance that participate in the flooding of traffic over point-to-multipoint LSPs are upgraded to Junos OS Release 8.3 or later to support this feature.

To flood unknown unicast, broadcast, and multicast traffic using point-to-multipoint LSPs, configure the **rsvp-te** statement as follows:

```
rsvp-te {
  label-switched-path-template {
    (default-template | lsp-template-name);
  }
  static-lsp lsp-name;
}
```

You can include this statement at the following hierarchy levels:

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

You can configure either a static point-to-multipoint LSP for VPLS flooding or a dynamic point-to-multipoint LSP.

NOTE: You cannot specify both the static and label-switched-path-template statements at the same time.

The following sections describe how to configure static and dynamic point-to-multipoint LSPs for flooding unknown traffic in a VPLS routing instance:

- Configuring Static Point-to-Multipoint Flooding LSPs on page 508
- Configuring Dynamic Point-to-Multipoint Flooding LSPs on page 509

Configuring Static Point-to-Multipoint Flooding LSPs

The **static-lsp** option creates a static flooding point-to-multipoint LSP that includes all of the neighbors in the VPLS routing instance. Flood traffic is sent to all of the VPLS neighbors using the generated point-to-multipoint LSP. VPLS neighbors added to the routing instance later are not automatically added to the point-to-multipoint LSP. You will need to manually add the new VPLS neighbors to the static point-to-multipoint flooding LSP. By configuring static point-to-multipoint LSPs for flooding, you have more control over which path each sub-LSP follows.

To configure a static flooding point-to-multipoint LSP, specify the name of the static flooding point-to-multipoint LSP by including 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]**

Configuring Dynamic Point-to-Multipoint Flooding LSPs

To configure a dynamic point-to-multipoint flooding LSP, include the **label-switched-path-template** statement option at the **[edit routing-instances *routing-instance-name* provider-tunnel rsvp-te]** hierarchy level:

```
[edit routing-instances routing-instance-name provider-tunnel rsvp-te]
label-switched-path-template {
  (default-template | lsp-template-name);
}
```

You can automatically generate the point-to-multipoint LSP to be used for flooding unknown traffic or you can manually configure the point-to-multipoint LSP:

- Configuring Dynamic Point-to-Multipoint Flooding LSPs with the Default Template on page 509
- Configuring Dynamic Point-to-Multipoint Flooding LSPs with a Preconfigured Template on page 510

Configuring Dynamic Point-to-Multipoint Flooding LSPs with the Default Template

The **default-template** option, specified at the **[edit routing-instances *routing-instance-name* provider-tunnel rsvp-te label-switched-path-template]** hierarchy level, causes the point-to-multipoint LSPs to be created with default parameters. The default parameters are for a minimally configured point-to-multipoint LSP. The name of this point-to-multipoint LSP is also generated automatically and is based on the following model:

id*:vpls:*router-id*:*routing-instance-name

The following **show** command output for **show mpls lsp p2mp** illustrates how a point-to-multipoint flood LSP name could appear if you configure the **label-switched-path-template** statement with the **default-template** option:

```
user@host> show mpls lsp p2mp ingress
Ingress LSP: 2 sessions P2MP name: static, P2MP branch count: 3
To              From            State Rt ActivePath       P     LSPname
10.255.14.181   10.255.14.172   Up     0                  *     vpn02-vpn11
10.255.14.177   10.255.14.172   Up     0 path2            *     vpn02-vpn07
10.255.14.174   10.255.14.172   Up     0 path3            *     vpn02-vpn04
P2MP name: 9:vpls:10.255.14.172:green, P2MP branch count: 2
To              From            State Rt ActivePath       P     LSPname
10.255.14.177   10.255.14.172   Up     0                  *
11:vpls:10.255.14.172:green
10.255.14.174   10.255.14.172   Up     0                  *
10:vpls:10.255.14.172:green
Total 5 displayed, Up 5, Down 0
```

The dynamically generated point-to-multipoint LSP name is **9:vpls:10.255.14.172:green**.

Configuring Dynamic Point-to-Multipoint Flooding LSPs with a Preconfigured Template

You can configure a point-to-multipoint flooding LSP template for the VPLS routing instance. The template allows you to specify the properties of the dynamic point-to-multipoint LSPs that are used to flood traffic for the VPLS routing instance. You can specify all of the standard options available for a point-to-multipoint LSP within this template. These properties are inherited by the dynamic point-to-multipoint flood LSPs.

To configure a point-to-multipoint LSP template for flooding VPLS traffic, specify all of the properties you want to include in a point-to-multipoint LSP configuration. To specify this LSP as a point-to-multipoint flooding template, include the **p2mp** and **template** statements:

```
p2mp;
template;
```

You can include these statements at the following hierarchy levels:

- **[edit protocols mpls label-switched-path *p2mp-lsp-template-name*]**
- **[edit logical-systems *logical-system-name* protocols mpls label-switched-path *p2mp-lsp-template-name*]**

For more information about how to configure the **p2mp** statement and point-to-multipoint LSPs, see the *Junos OS MPLS Applications Configuration Guide*.

Once you have configured the point-to-multipoint LSP template, specify the name of the point-to-multipoint LSP template with the **label-switched-path-template** statement:

```
label-switched-path-template p2mp-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]**

Configuring VPLS and Integrated Routing and Bridging

Traditional Layer 2 switching environments consist of Layer 2 devices (such as switches) that partition data into broadcast domains. The broadcast domains can be created through physical topologies or logically through virtual local area networks (VLANs). For MX Series routers, you can logically configure broadcast domains within virtual switch routing instances, VPLS routing instances, or bridging domains. The individual routing instances or bridging domains are differentiated through VLAN identifiers and these instances or domains function much like traditional VLANs.

For detailed information and configuration instructions on bridging domains and spanning tree protocol, see the *Junos OS Network Interfaces Configuration Guide*, the *Junos OS Routing Protocols Configuration Guide*, and the *Junos OS Feature Guide*.

The following sections provide configuration information specific to VPLS in regards to integrated routing and bridging:

Configuring MAC Address Flooding and Learning for VPLS

In a VPLS routing instance or bridge domain, when a frame is received from a CE interface, it is flooded to the other CE interfaces and all of the VE interfaces if the destination MAC address is not learned or if the frame is either broadcast or multicast. If the destination MAC address is learned on another CE device, such a frame is unicasted to the CE interface on which the MAC address is learned. This might not be desirable if the service provider does not want CE devices to communicate with each other directly.

To prevent CE devices from communicating directly include the **no-local-switching** statement at the **[edit bridge-domains *bridge-domain-name*]** hierarchy level:

```
[edit bridge-domains bridge-domain-name]
no-local-switching;
```

The **no-local-switching** statement is available only on MX Series routers. If you include it, frames arriving on a CE interface are sent to VE or core-facing interfaces only.

Configuring MSTP for VPLS

When you configure integrated routing and bridging, you might also need to configure the Multiple Spanning Tree Protocol (MSTP). When you configure MSTP on a provider edge (PE) router running VPLS, you must also configure **ethernet-vpls** encapsulation on the customer-facing interfaces. VLAN-based VPLS interface encapsulations are not supported with MSTP.

Configuring Interoperability Between BGP Signaling and LDP Signaling in VPLS

A single VPLS routing instance can encompass one set of PE routers that use BGP for signaling and another set of PE routers that use LDP for signaling. Within each set, all of the PE routers are fully meshed in both the control and data planes and have a bidirectional pseudowire to each of the other routers in the set. However, the BGP-signaled routers cannot be directly connected to the LDP-signaled routers. To be able to manage the two separate sets of PE routers in a single VPLS routing instance, a border PE router must be configured to interconnect the two sets of routers.

The VPLS RFCs and Internet drafts require that all of the PE routers participating in a single VPLS routing instance must be fully meshed in the data plane. In the control plane, each fully meshed set of PE routers in a VPLS routing instance is called a PE router mesh group. The border PE router must be reachable by and have bidirectional pseudowires to all of the PE routers that are a part of the VPLS routing instance, both the LDP-signaled and BGP-signaled routers.

For LDP BGP interworking to function, LDP-signaled routers can only be configured with forwarding equivalence class (FEC) 128.

The following sections describe how to configure BGP LDP interworking for VPLS:

LDP BGP Interworking Platform Support

LDP BGP interworking is supported on the following Juniper Networks routers and routing platforms:

- M7i
- M10i
- M40e
- M120
- M320
- MX Series routers
- T Series routers
- TX Matrix routers

Configuring VPLS Mesh Groups for LDP BGP Interworking

To configure LDP BGP interworking for VPLS, include the **mesh-group** statement in the VPLS routing instance configuration of the PE border router:

```
mesh-group mesh-group-name {
    local-switching;
    mac-flush [ explicit-mac-flush-message-options ];
    neighbor address;
    peer-as all;
    vpls-id number;
}
```

You can include this statement at the following hierarchy levels:

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

Using the **neighbor** statement, configure each PE router that is a part of the mesh group. You must separate the LDP-signaled routers and the BGP-signaled routers into their own respective mesh groups. The LDP-signaled routers can be divided into multiple mesh groups. The BGP-signaled routers must be configured within a single mesh group for each routing instance.

Configuring Switching Between Pseudowires Using VPLS Mesh Groups

To configure switching between Layer 2 circuit pseudowires using VPLS mesh groups, you can do either of the following:

- Configure a mesh group for each Layer 2 circuit pseudowire terminating at a VPLS routing instance. You can configure a maximum of 16 mesh groups on MX Series routers and a maximum of 256 mesh groups for M Series and T Series routers.
- Configure a single mesh group, terminate all the Layer 2 circuit pseudowires into it, and enable local switching between the pseudowires by including the **local-switching** statement at the **[edit routing-instances *routing-instance-name* protocols vpls mesh-group *mesh-group-name*]** hierarchy level. By default, you cannot configure local switching for mesh groups (except for the CE mesh group) because all of the VPLS PE routers must be configured in a full mesh. However, local switching is useful if you are terminating Layer 2 circuit pseudowires in a mesh group configured for an LDP signaled VPLS routing instance.

NOTE: Do not include the local-switching statement on PE routers configured in a full mesh VPLS network.

To terminate multiple pseudowires at a single VPLS mesh group, include the **local-switching** statement:

```
local-switching;
```

You can include this statement at the following hierarchy levels:

- **[edit routing-instances *routing-instance-name* protocols vpls mesh-group *mesh-group-name*]**
- **[edit logical-systems *logical-system-name* routing-instances *routing-instance-name* protocols vpls mesh-group *mesh-group-name*]**

Configuring Integrated Routing and Bridging Support for LDP BGP Interworking with VPLS

Beginning with Junos OS Release 9.4, you can configure an integrated routing and bridging (IRB) interface on a router that functions as an autonomous system border router (ASBR) in an inter-AS VPLS environment between BGP-signaled VPLS and LDP-signaled VPLS. Previously, IRB interfaces were supported only on Provider Edge (PE) routers.

To configure a IRB support for LDP BGP Interworking with VPLS, include the **routing-interface *interface-name*** statement.

You can include this statement at the following hierarchy levels:

- [edit routing-instances *routing-instance-name*]
- [edit logical-routers *logical-router-name* routing-instances *routing-instance-name*]

Configuring Inter-AS VPLS with MAC Processing at the ASBR

Inter-AS VPLS with MAC processing at the ASBR enables you to interconnect customer sites that are located in different ASs. In addition, you can configure the ASs with different signaling protocols. You can configure one of the ASs with BGP-signaled VPLS and the other with LDP-signaled VPLS. For more information about how to configure LDP-signaled and BGP signaled VPLS, see "Configuring Interoperability Between BGP Signaling and LDP Signaling in VPLS" on page 511.

For inter-AS VPLS to function properly, you need to configure IBGP peering between the PE routers, including the ASBRs in each AS, just as you do for a typical VPLS configuration. You also need to configure EBGP peering between the ASBRs in the separate ASs. The EBGP peering is needed between the ASBRs only. The link between the ASBR routers does not have to be Ethernet. You can also connect a CE router directly to one of the ASBRs, meaning you do not have to have a PE router between the ASBR and the CE router.

The configuration for the connection between the ASBRs makes inter-AS VPLS with MAC operations unique. The other elements of the configuration are described in other sections of this manual. An extensive configuration example for inter-AS VPLS with MAC operations is provided in the *Junos OS Feature Guide*.

The following sections describe how to configure inter-AS VPLS with MAC operations:

Inter-AS VPLS with MAC Operations Configuration Summary

This section provides a summary of all of the elements which must be configured to enable inter-AS VPLS with MAC operations. These procedures are described in detail later in this chapter and in other parts of the *Junos OS VPNs Configuration Guide*.

The following lists all of major elements of an inter-AS VPLS with MAC operations configuration:

- Configure IBGP between all of the routers within each AS, including the ASBRs.
- Configure EBGP between the ASBRs in the separated ASs. The EBGP configuration includes the configuration that interconnects the ASs.
- Configure a full mesh of LSPs between the ASBRs.
- Configure a VPLS routing instance encompassing the ASBR routers. The ASBRs are VPLS peers and are linked by a single pseudowire. Multihoming between ASs is not supported. A full mesh of pseudowires is needed between the ASBR routers in all of the interconnected ASs.
- Configure the VPLS routing instances using either BGP signaling or LDP signaling. LDP BGP interworking is supported for inter-AS VPLS with MAC operations, so it is possible

to interconnect the BGP-signaled VPLS routing instances with the LDP-signaled VPLS routing instances.

- Configure a single VPLS mesh group for all of the ASBRs interconnected using inter-AS VPLS.

Configuring the ASBRs for Inter-AS VPLS

This section describes the configuration on the ASBRs needed to enable inter-AS VPLS with MAC operations.

On each ASBR, you need to configure a VPLS mesh group within the VPLS routing instance which needs to include all of the PE routers within the AS, in addition to the ASBR. You need to configure the same mesh group for each of the ASs you want to interconnect using inter-AS VPLS. The mesh group name should be identical on each AS. You also must include the **peer-as all** statement. This statement enables the router to establish a single pseudowire to each of the other ASBRs.

To configure the mesh group on each ASBR, include the **mesh-group** and **peer-as all** statements:

```
mesh-group mesh-group-name {
    peer-as all;
}
```

You can include these statements at the following hierarchy levels:

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

Configuring Ingress Replication for IP Multicast Using Next Gen MVPN

Requirements

The routers used in this example are Juniper Networks M Series Multiservice Edge Routers, T Series Core Routers, or MX Series 3D Universal Edge Routers. When using ingress replication for IP multicast, each participating router must be configured with BGP 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, depending on the configuration of the provider tunnel in the routing instance.

Overview

The **ingress-replication** provider tunnel type uses unicast tunnels between routers to create a multicast distribution tree.

The **mpls-internet-multicast** routing instance type uses ingress replication provider tunnels to carry IP multicast data between routers through an MPLS cloud, using MBGP (or Next Gen) MVPN. Ingress replication can also be configured when using MVPN to carry multicast data between PE routers.

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 IP multicast are associated only with the default routing instance (**inet.0**), not with a configured routing instance. The **mpls-internet-multicast** routing instance type is configured for the default master instance on each router, and is also included at the **[edit protocols pim]** hierarchy level in the default instance.

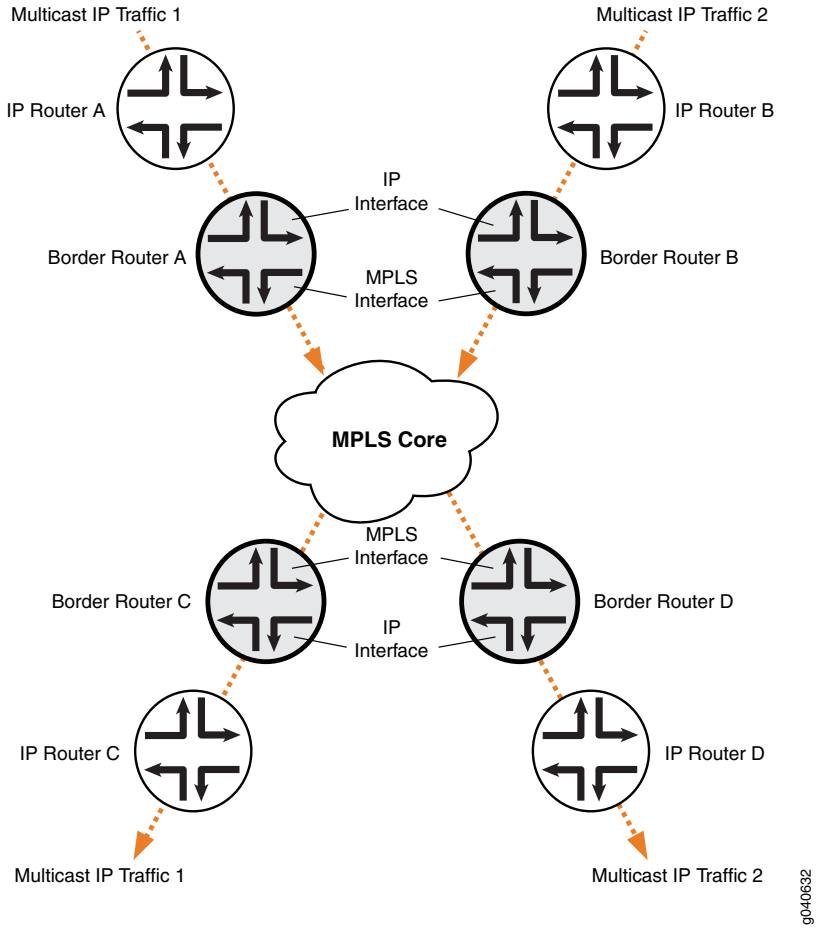
For each **mpls-internet-multicast** routing instance, the **ingress-replication** statement is required under the **provider-tunnel** statement and also under the **[edit routing-instances *routing-instance-name* provider-tunnel selective group source]** hierarchy level.

When a new destination needs to be added to the ingress replication provider tunnel, the resulting behavior differs depending on which mode has been configured for the tunnel:

- **existing-unicast-tunnel**—In this default mode, an existing unicast tunnel to the destination is used. If a unicast tunnel is not available, the destination is not added. This is the only mode available when using LDP LSPs and ingress replication.
- **create-new-ucast-tunnel**—When this mode 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.

The IP topology consists of routers on the edge of the IP multicast domain. Each router has a set of IP interfaces configured toward the MPLS cloud and a set of interfaces configured toward the IP routers. See Figure 57 on page 517. 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 IBGP session for the control plane.

Figure 57: Internet Multicast Topology



Configuration

CLI Quick Configuration

Experienced users can copy and paste the CLI statements below to quickly configure an IP **mpls-internet-multicast** routing instance named IM-A with ingress replication provider tunnels that creates a new unicast tunnel to each new destination requested. Be sure to replace ***group-address*** and ***source-address*** with the correct IP address values for your system.

```
[edit]
set routing-instances IM-A instance-type mpls-internet-multicast
set routing-instances IM-A provider-tunnel ingress-replication create-new-ucast-tunnel
set routing-instances IM-A provider-tunnel ingress-replication label-switched-path
  label-switched-path-template default-template
set routing-instances IM-A provider-tunnel selective group group-address source
  source-address ingress-replication label-switched-path
set routing-instances IM-A protocols mvpn
set protocols pim mpls-internet-multicast
```

Step-by-Step Procedure

Configure IP Multicast Using Ingress Replication Tunnels

The following example shows how to configure ingress replication on IP multicast instance **IM-A** with the routing instance type **mpls-internet-multicast**. Additionally, this example shows how to configure a selective provider tunnel that selects a new unicast tunnel each time a new destination needs to be added to the multicast distribution tree.

1. Configure the routing instance type for IM-A to be **mpls-internet-multicast**.

```
[edit]
user@host# edit routing-instances
[edit routing-instances]
user@host# set IM-A instance-type mpls-internet-multicast
```

2. Configure the ingress replication provider tunnel to create a new unicast tunnel each time a destination needs to be added to the multicast distribution tree.

```
[edit routing-instances]
user@host# set IM-A provider-tunnel ingress-replication create-new-ucast-tunnel
```

NOTE: Alternatively, use the **existing-unicast-tunnel** statement if an existing tunnel should be used each time a destination needs to be added. It is the only mode available when using LDP LSPs and ingress replication.

3. Configure the point-to-point LSP to use the default template settings (this is needed only when using RSVP tunnels).

```
[edit routing-instances]
user@host# set IM-A provider-tunnel ingress-replication label-switched-path
label-switched-path-template default-template
```

4. Configure selective ingress replication provider tunnels.

```
[edit routing-instances]
user@host# set IM-A provider-tunnel selective group 232.1.1.1/32 source
192.168.195.145/32 ingress-replication label-switched-path
```

5. Configure the MVPN Protocol in the routing instance.

```
[edit routing-instances]
user@host# set IM-A protocols mvpn
user@host# up
```

6. Add the **mpls-internet-multicast** configuration statement under the **[protocols pim]** hierarchy level in the master instance.

```
[edit]
user@host# edit protocols
```

```
[edit protocols]

user@host# set pim mpls-internet-multicast

user@host# top
```

7. Commit the configuration.

```
[edit]

user@host# commit
```

8. Use the **show ingress-replication mvpn** command to check the ingress replication status.

```
[edit]

user@host# run show ingress-replication mvpn

Ingress Tunnel: mvpn:1
  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
```

9. Use the **show mvpn instance** command to show the ingress replication tunnel type.

```
[edit]

user@host# run show mvpn instance IM-A

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 : IM-A
   MVPN Mode : SPT-ONLY
   Provider tunnel: S-P-tnl:INGRESS-REPLICATION:MPLS Label 18:10.255.245.6
   Neighbor               S-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
```

Related Documentation

- Configuring Routing Instances for an MBGP MVPN on page 391
- **mpls-internet-multicast on page 432**
- **ingress-replication on page 430**
- **create-new-ucast-tunnel on page 421**
- **existing-unicast-tunnel on page 422**
- show ingress-replication mvpn

Tracing VPLS Traffic and Operations

To trace VPLS traffic, include the **traceoptions** statement:

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

The following trace flags display the operations associated with VPLS:

- **all**—All VPLS tracing options
- **connections**—VPLS connections (events and state changes)
- **error**—Error conditions
- **nlri**—VPLS advertisements received or sent using BGP
- **route**—Trace-routing information
- **topology**—VPLS topology changes caused by reconsideration or advertisements received from other PE routers using BGP

Configuring Port Mirroring for VPLS Traffic

You can configure port mirroring for VPLS traffic on the M7, M10i, M120, M320, and the MX Series routers. VPLS port mirroring is supported only M7i and M0i routers with the Enhanced Compact Forwarding Engine Board (CFEB-E). In addition, on M320 routers, VPLS port mirroring is supported only on Enhanced III Flexible PIC Concentrators (FPCs).

To configure port mirroring for VPLS include the **port-mirroring** statement at the **[edit forwarding-options]** hierarchy level. For more information about configuring port mirroring for VPLS for all platforms supported, see the *Junos OS Policy Framework Configuration Guide*. For information about configuring port mirroring for VPLS for MX Series routers, see the *JUNOS MX Series Ethernet Services Routers Layer 2 Configuration Guide*.

Configuring the Label Block Size

VPLS MPLS packets have a two-label stack. The outer label is used for normal MPLS forwarding in the service provider's network. If BGP is used to establish VPLS, the inner label is allocated by a PE router as part of a label block. One inner label is needed for each remote VPLS site. Four sizes are supported. We recommend using the default size of 8, unless the network design requires a different size for optimal label usage, to allow the router to support a larger number of VPLS instances.

If you allocate a large number of small label blocks to increase efficiency, you also increase the number of routes in the VPLS domain. This has an impact on the control plane overhead.

Changing the configured label block size causes all existing pseudowires to be deleted. For example, if you configure the label block size to be 4 and then change the size to 8, all existing label blocks of size 4 are deleted, which means that all existing pseudowires are deleted. The new label block of size 8 is created, and new pseudowires are established.

Four label block sizes are supported: 2, 4, 8, and 16. Consider the following scenarios:

- 2—Allocate the label blocks in increments of 2. For a VPLS domain that has only two sites with no future expansion plans.
- 4—Allocate the label blocks in increments of 4.
- 8 (default)—Allocate the label blocks in increments of 8.
- 16—Allocate the label blocks in increments of 16. A label block size of 16 enables you to minimize the number of routes in the VPLS domain. Use this setting only if the number of routes is the most important concern.

Configure the label block size:

```
[edit routing-instances instance-name protocols vpls]
user@router# set label-block-size 2
```

Related Documentation

- Configuring VPLS Routing Instances on page 473

Configuring Y.1731 Functionality for VPLS to Support Delay and Delay Variation

For VPLS, you can configure the Ethernet frame delay measurement (ETH-DM) functionality to trigger two-way ETH-DM and allow concurrent ETH-DM CLI sessions from the same local maintenance association end point (MEP). The feature also provides the option to perform ETH-DM for a given 802.1q priority, to set the size of the data type, length, and value (TLV), to disable the **session-id-tlv** option, and to generate XML output.

This feature complements the ITU-T Y.1731 Ethernet service OAM feature. On-demand delay measurement for VPLS is supported on MX Series routers installed with Rev-B DPCs. Only the two-way delay measurement feature is supported for VPLS connections.

This feature is currently supported only for up MEPs. Set the MEP direction to up by configuring the **up** option for the **direction** statement at the **[edit protocols oam ethernet**

connectivity-fault-management maintenance-domain *md-name* **maintenance-association** *ma-name* **mep** *mep-id***]** hierarchy level.

This feature also provides support for an optional configuration where you can delegate the server-side processing (for two-way delay measurement) to the Packet Forwarding Engine (PFE) to prevent overloading on the Routing Engine. To enable this feature, include the **delegate-server-processing** statement at the **[edit protocols oam Ethernet connectivity-fault-management performance-monitoring]** hierarchy level. By default, the server-side processing is done by the Routing Engine.

The following commands enable you to monitor and maintain the Y.1731 feature for VPLS:

- To display the delay measurement values across a VPLS connection, use the **monitor ethernet delay-measurement two-way (***remote-mac-address* **| mep** *mep-id***) maintenance-domain** *name* **maintenance-association** *name* **count** *count* **wait** *time* **priority 802.1p-value** *size* **no-session-id-tlv xml** command.
- The feature also provides support for enhanced continuity measurement by using an existing continuity check protocol. The continuity for every remote MEP is measured as the percentage of time that a remote MEP was operationally up over the total administratively enabled time.

  To display the continuity measurement information, use the **show oam ethernet connectivity-fault-management mep-database maintenance-domain** *name* **maintenance-association** *name* **local-mep** *identifier* **remote-mep** *identifier* command.
- You can restart the continuity measurement by clearing the currently measured operational uptime and administrative enabled time. To clear the existing continuity measurement and restart counting the operational uptime, use the **clear oam ethernet connectivity-fault-management continuity-measurement maintenance-domain** *name* **maintenance-association** *name* **local-mep** *identifier* **remote-mep** *identifier* command.
- To clear the delay statistics, issue a **clear oam ethernet connectivity-fault-management statistics** command or a **clear oam ethernet connectivity-fault-management delay-statistics two-way maintenance-domain** *md-name* **maintenance-association** *ma-name* command.

Related Documentation

- ITU-T Y.1731 Ethernet Service OAM
- Configuring MEP Interfaces to Support Ethernet Frame Delay Measurements
- Example: Configuring Two-Way Ethernet Frame Delay Measurements with Single-Tagged Interfaces

CHAPTER 20

VPLS Example

Example: Building a VPLS From Router 1 to Router 3

This example illustrates how VPLS label blocks are allocated for a specific configuration. It is organized in the following sections:

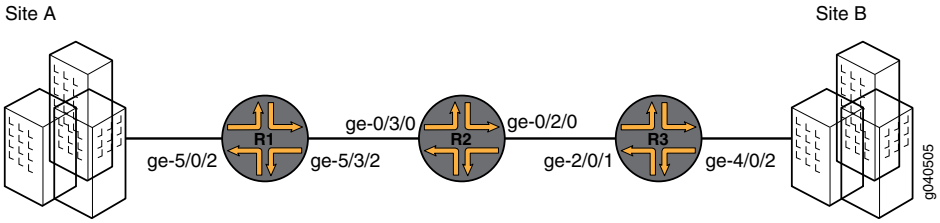
Requirements

This configuration example requires three Juniper Networks routers.

Overview and Topology

In the network shown in Figure 58 on page 523 Router 1 is establishing a pseudowire to Router 3

Figure 58: Router 1 to Router 3 Topology



Each PE filters the VPLS NLRI contained in the BGP update messages based on route target communities. Those VPLS NLRI instances that match the route target (in this case 8717:2000:2:1) are imported for further processing. The NLRI for Router 1 and Router 3 is shown in Table 12 on page 523.

Table 12: NLRI Exchange Between for Router 1 and Router 3

Router 1 NLRI Advertisement to Router 3	Router 3 NLRI Advertisement to Router 1
RD - 8717:1000	RD - 8717:1000
VE ID - 1	VE ID - 2

Table 12: NLRI Exchange Between for Router 1 and Router 3 *(continued)*

Router 1 NLRI Advertisement to Router 3	Router 3 NLRI Advertisement to Router 1
VE Block Offset - 1	VE Block Offset - 1
VE Block Size - 8	VE Block Size - 8
Label Base - 262161	Label Base - 262153

To set up a pseudowire to Router 3, Router 1 must select a label to use to send traffic to Router 3 and also select a label that it expects Router 3 to use to send traffic to itself. The site ID contained in the VPLS NLRI from Router 3 is 2.

Router 1 learns of the existence of site ID 2 in the same VPLS domain. Using the equation VBO <= Local Site ID < (VBO +VBS), Router 1 checks if the route advertised by site ID 2 fits in the label block and block offset that it previously advertised to Router 3. In this example it does fit, so the site ID 2 is mapped by the VPLS NLRI advertised by Router 1, and Router 1 is ready to set up a pseudowire to Router 3.

To select the label to reach Router 3, Router 1 looks at the label block advertised by Router 3 and performs a calculation. The calculation a PE router uses to check if its site ID is mapped in the label block from the remote peer is VBO <= Local Site ID < (VBO +VBS). So, Router 1 selects label (262153 + (1 -1)) = 262153 to send traffic to Router 3. Using the same equation, Router 1 looks at its own label block that it advertised and selects label (262161 + (2 - 1)) = 262162 to receive traffic from Router 3. Router 1 programs its forwarding state such that any traffic destined to Router 3 carries the pseudowire label 262153 and any traffic coming from Router 3 is expected to have the pseudowire label 262162. This completes the operations on the VPLS NLRI received from Router 3. Router 1 now has a pseudowire set up to Router 3.

Router 3 operation is very similar to the Router 1 operation. Since the Router 3 site ID of 2 fits in the label block and block offset advertised by Router 1, Router 3 selects label (262161 + (2 - 1)) = 262162 to send traffic to Router 1. Router 3 looks at its own label block that it advertised and selects label (262153 + (1 - 1)) = 262153 to receive traffic from Router 1. This completes the creation of a pseudowire to Router 1.

By default, for VPLS operation Junos OS uses a virtual tunnel (VT) loopback interface to represent a pseudowire. This example uses a label-switched interface (LSI) instead of a VT interface because there is no change in the VPLS control plane operation. Thus, for an MX platform, if there is a tunnel physical interface card (PIC) configured, it is mandatory to include the **no-tunnel-services** statement at the **[edit routing-instances *routing-instance-name* protocols vpls]** hierarchy level.

Configuration

The following sections present the steps to configure and verify the example in Figure 58 on page 523.

- Configuring Router 1 on page 525
- Configuring Router 3 on page 525
- Verifying the VPLS Label Allocations on page 526

Configuring Router 1

Step-by-Step Procedure

1. Configure Router 1. Create the **edut** routing instance. Specify the **vpls** instance type. Configure the route distinguisher and specify the value **8717:1000**. Configure the route target and specify the value **8717:100** Configure the VPLS protocol. Specify **10** as the site range. Specify **1** as the site ID. Include the **no-tunnel-services** statement.

```
[edit routing-instances]
edut {
   instance-type vpls;
   interface ge-5/0/2.0;
   route-distinguisher 8717:1000;
   vrf-target target:8717:100;
   protocols {
      vpls {
         site-range 10;
         no-tunnel-services;
         site router-1 {
            site-identifier 1;
         }
      }
   }
}
```

Configuring Router 3

Step-by-Step Procedure

1. Configure Router 3. Create the **edut** routing instance. Specify the **vpls** instance type. Configure the route distinguisher and specify the value **8717:2000**. Configure the route target and specify the value **8717:200** Configure the VPLS protocol. Specify **10** as the site range. Specify **2** as the site ID. Include the **no-tunnel-services** statement.

```
[edit routing-instances]
edut {
   instance-type vpls;
   interface ge-4/0/2.0;
   route-distinguisher 8717:2000;
   vrf-target target:8717:100;
   protocols {
      vpls {
         site-range 10;
         no-tunnel-services;
         site router-3 {
            site-identifier 2;
         }
      }
```

```
    }
  }
```

Verifying the VPLS Label Allocations

Step-by-Step Procedure

1. As shown in the figure and the configuration, Site A is attached to Router 1. Site A is assigned a site ID of 1. Before Router 1 can announce its membership to VPLS **edut** using a BGP update message, Router 1 needs to allocate a default label block. In this example, the label base of the label block allocated by Router 1 is 262161. Since Router 1's site ID is 1, Router 1 associates the assigned label block with block offset of 1. The following messages aresent from Router 1 to Router 3 and displayed using the **monitor traffic interface *interface-name*** command:

```
user@Router1> monitor traffic interface ge-5/3/2

Jun 14 12:26:31.280818 BGP SEND 10.10.10.1+179 -> 10.10.10.3+53950
Jun 14 12:26:31.280824 BGP SEND message type 2 (Update) length 88
Jun 14 12:26:31.280828 BGP SEND flags 0x40 code Origin(1): IGP
Jun 14 12:26:31.280833 BGP SEND flags 0x40 code ASPath(2) length 0: <null>
Jun 14 12:26:31.280837 BGP SEND flags 0x40 code LocalPref(5): 100
Jun 14 12:26:31.280844 BGP SEND flags 0xc0 code Extended Communities(16): 2:8717:100 800a:19:0:0
Jun 14 12:26:31.280848 BGP SEND flags 0x90 code MP_reach(14): AFI/SAFI 25/65
Jun 14 12:26:31.280853 BGP SEND         nhop 10.10.10.1 len 4
Jun 14 12:26:31.280862 BGP SEND         8717:1000:1:1 (label base : 262161 range : 8, ce id: 1, offset: 1)
Jun 14 12:26:31.405067 BGP RECV 10.10.10.3+53950 -> 10.10.10.1+179
Jun 14 12:26:31.405074 BGP RECV message type 2 (Update) length 88
Jun 14 12:26:31.405080 BGP RECV flags 0x40 code Origin(1): IGP
Jun 14 12:26:31.405085 BGP RECV flags 0x40 code ASPath(2) length 0: <null>
Jun 14 12:26:31.405089 BGP RECV flags 0x40 code LocalPref(5): 100
Jun 14 12:26:31.405096 BGP RECV flags 0xc0 code Extended Communities(16): 2:8717:100 800a:19:0:0
Jun 14 12:26:31.405101 BGP RECV flags 0x90 code MP_reach(14): AFI/SAFI 25/65
Jun 14 12:26:31.405106 BGP RECV         nhop 10.10.10.3 len 4
Jun 14 12:26:31.405116 BGP RECV         8717:2000:2:1 (label base : 262153 range : 8, ce id: 2, offset:
1)
```

2. As shown in the figure and the configuration, Site B is attached to Router 3. Site B is assigned a site ID of 2. Before Router 3 can announce its membership to VPLS **edut** using a BGP update message, Router 3 assigns a default label block with the label base of **262153**. The block offset for this label block is 1 because its own site ID of 2 fits in the block being advertised. The following messages are sent from Router 3 to Router 1 and displayed using the **monitor traffic interface *interface-name*** command:

```
user@Router3> monitor traffic interface ge-2/0/1

Jun 14 12:26:31.282008 BGP SEND 10.10.10.3+53950 -> 10.10.10.1+179
Jun 14 12:26:31.282018 BGP SEND message type 2 (Update) length 88
Jun 14 12:26:31.282026 BGP SEND flags 0x40 code Origin(1): IGP
Jun 14 12:26:31.282034 BGP SEND flags 0x40 code ASPath(2) length 0: <null>
Jun 14 12:26:31.282041 BGP SEND flags 0x40 code LocalPref(5): 100
Jun 14 12:26:31.282052 BGP SEND flags 0xc0 code Extended Communities(16): 2:8717:100 800a:19:0:0
Jun 14 12:26:31.282078 BGP SEND flags 0x90 code MP_reach(14): AFI/SAFI 25/65
Jun 14 12:26:31.282088 BGP SEND         nhop 10.10.10.3 len 4
Jun 14 12:26:31.282102 BGP SEND         8717:2000:2:1 (label base : 262153 range : 8, ce id: 2, offset: 1)

Jun 14 12:26:31.283395 BGP RECV 10.10.10.1+179 -> 10.10.10.3+53950
Jun 14 12:26:31.283405 BGP RECV message type 2 (Update) length 88
Jun 14 12:26:31.283412 BGP RECV flags 0x40 code Origin(1): IGP
```

```
Jun 14 12:26:31.283419 BGP RECV flags 0x40 code ASPath(2) length 0: <null>
Jun 14 12:26:31.283426 BGP RECV flags 0x40 code LocalPref(5): 100
Jun 14 12:26:31.283435 BGP RECV flags 0xc0 code Extended Communities(16): 2:8717:100 800a:19:0:0
Jun 14 12:26:31.283443 BGP RECV flags 0x90 code MP_reach(14): AFI/SAFI 25/65
Jun 14 12:26:31.283471 BGP RECV         nhop 10.10.10.1 len 4
Jun 14 12:26:31.283486 BGP RECV         8717:1000:1:1 (label base : 262161 range : 8, ce id: 1, offset:
1)
```

3. Verify the connection status messages for Router 1 using the **show vpls connections** command. Notice the base label is **262161**, the incoming label from Router 3 is **262162**, and the outgoing label to Router 3 is **262153**.

```
user@Router1> show vpls connections instance edut extensive

Instance: edut
  Local site: router-1 (1)
    Number of local interfaces: 1
    Number of local interfaces up: 1
    IRB interface present: no
    ge-5/0/2.0
    lsi.1049600         2         Intf - vpls edut local site 1 remote site 2
    Label-base        Offset     Range     Preference
    262161            1          8         100
    connection-site           Type  St     Time last up          # Up trans
    2                         rmt   Up     Jun 14 12:26:31 2009           1
      Remote PE: 10.10.10.3, Negotiated control-word: No
      Incoming label: 262162, Outgoing label: 262153
      Local interface: lsi.1049600, Status: Up, Encapsulation: VPLS
        Description: Intf - vpls edut local site 1 remote site 2
    Connection History:
        Jun 14 12:26:31 2009  status update timer
        Jun 14 12:26:31 2009  loc intf up                  lsi.1049600
        Jun 14 12:26:31 2009  PE route changed
        Jun 14 12:26:31 2009  Out lbl Update                    262153
        Jun 14 12:26:31 2009  In lbl Update                     262162
        Jun 14 12:26:31 2009  loc intf down

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 availble
BK -- Backup connection          ST -- Standby connection
PF -- Profile parse failure      PB -- Profile busy

Legend for interface status
```

```
Up -- operational
Dn -- down
```

4. Verify the connection status messages for Router 3 using the **show vpls connections** command. Notice the base label is **262153**, the incoming label from Router 1 is **262153**, and the outgoing label to Router 1 is **262162**.

```
user@Router3> show vpls connections instance edut extensive

Instance: edut
  Local site: router-3 (2)
    Number of local interfaces: 1
    Number of local interfaces up: 1
    IRB interface present: no
    ge-4/0/2.0
    lsi.1050368         1         Intf - vpls edut local site 2 remote site 1
    Label-base        Offset     Range     Preference
    262153            1          8         100
    connection-site           Type  St     Time last up          # Up trans
    1                         rmt   Up     Jun 14 12:26:31 2009           1
      Remote PE: 10.10.10.1, Negotiated control-word: No
      Incoming label: 262153, Outgoing label: 262162
      Local interface: lsi.1050368, Status: Up, Encapsulation: VPLS
        Description: Intf - vpls edut local site 2 remote site 1
    Connection History:
        Jun 14 12:26:31 2009  status update timer
        Jun 14 12:26:31 2009  loc intf up                  lsi.1050368
        Jun 14 12:26:31 2009  PE route changed
        Jun 14 12:26:31 2009  Out lbl Update                    262162
        Jun 14 12:26:31 2009  In lbl Update                     262153
        Jun 14 12:26:31 2009  loc intf down

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 availble
BK -- Backup connection          ST -- Standby connection
PF -- Profile parse failure      PB -- Profile busy

Legend for interface status
Up -- operational
Dn -- down
```

Related Documentation

- VPLS Label Blocks Operation on page 464

CHAPTER 21

Summary of VPLS Configuration Statements

active-interface

Syntax

```
active-interface {
    any;
    primary interface-name;
}
```

Hierarchy Level

[edit logical-systems *logical-system-name* routing-instances *routing-instance-name* protocols vpls site *site-name*],
[edit routing-instances *routing-instance-name* protocols vpls site *site-name*]

Release Information

Statement introduced in Junos OS Release 7.5.

Description

Specify a multihomed interface as the primary interface for the VPLS site. If there are multiple interfaces, the remaining interfaces are activated only when the primary interface goes down. If no active interfaces are configured at the site level, it is assumed that all traffic for a VPLS site travels through a single, nonmultihomed PE router.

Options

any—One configured interface is randomly designated as the active interface for the VPLS site.

primary *interface-name*—Specify the name of the multihomed interface to be used as the primary interface by the VPLS site.

Required Privilege Level

routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation

- Specifying an Interface as the Active Interface on page 506

automatic-site-id

Syntax

```
automatic-site-id {
  collision-detect-time seconds;
  new-site-wait-time seconds;
  reclaim-wait-time minimum seconds maximum seconds;
  startup-wait-time seconds;
}
```

Hierarchy Level

[edit logical-systems *logical-system-name* routing-instances *routing-instance-name* protocols vpls site *site-name*],
[edit routing-instances *routing-instance-name* protocols vpls site *site-name*]

Release Information

Statement introduced in Junos OS Release 9.1.

Description

Enable automatic site identifiers for VPLS routing instances.

Options

collision-detect-time—The time in seconds to wait after a claim advertisement is sent to the other routers in a VPLS instance before a PE router can begin using a site identifier. If the PE router receives a competing claim advertisement for the same site identifier during this time period, it initiates the collision resolution procedure for site identifiers.

new-site-wait-time—The time in seconds to wait to receive VPLS information for a newly configured routing instance or a new site. This time interval is also applied whenever the automatic site identifier feature is activated on a VPLS routing instance other than at startup. Effectively, this timer indicates how long to wait before an attempt is made to allocate a site identifier. This timer is also triggered whenever a VPLS routing instance is enabled.

reclaim-wait-time—The time in seconds to wait to receive VPLS information for a newly configured routing instance or a new site. This time interval is also applied whenever the automatic site identifier feature is activated on a VPLS routing instance other than at startup. Effectively, this timer indicates how long to wait before an attempt is made to allocate a site identifier. This timer is also triggered whenever a VPLS routing instance is enabled. You can configure two values for this option: the **minimum** wait time and the **maximum** wait time.

startup-wait-time—The time in seconds to wait at startup to receive all the VPLS information for the route targets configured on the other PE routers included in the VPLS routing instance.

Required Privilege Level

routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation

- Configuring Automatic Site Identifiers for VPLS on page 476

connectivity-type

Syntax connectivity-type (ce | irb | permanent);

Hierarchy Level [edit logical-systems *logical-system-name* routing-instances *routing-instance-name* protocols vpls],
[edit routing-instances *routing-instance-name* protocols vpls]

Release Information Statement introduced in Junos OS Release 9.1.
irb option introduced in Junos OS Release 9.3.
permanent option introduced in Junos OS Release 10.4.

Description Specify when a VPLS connection is taken down depending on whether or not the interface for the VPLS routing instance is customer-facing or integrated routing and bridging (IRB).

Default ce

Options

ce—Require that for the VPLS connection to be up, the customer-facing interface for the VPLS routing instance must also be up. If the customer-facing interface fails, the VPLS connection is taken down.

irb—Allow a VPLS connection to remain up so long as an IRB interface is configured for the VPLS routing instance.

permanent—Allow a VPLS connection to remain up until specifically taken down. This option is reserved for use in configuring Layer 2 Wholesale subscriber networks. See the *Broadband Subscriber Management Solutions Guide* for details about configuring a Layer 2 Wholesale network.

NOTE: To specifically take down a VPLS routing instance that is using the permanent option, all associated static logical interfaces must also be down.

Required Privilege Level routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation

- Configuring VPLS Routing Instance and VPLS Interface Connectivity on page 480
- Configuring Separate Routing Instances for Layer 2 Wholesale Service Retailers

encapsulation

Syntax encapsulation (ethernet-vpls | ether-vpls-over-atm-llc | extended-vlan-vpls | vlan-vpls);

Hierarchy Level [edit interfaces *interface-name*],
[edit logical-systems *logical-system-name* interfaces *interface-name*]

Release Information Statement introduced before Junos OS Release 7.4.

Description Physical link-layer encapsulation type for VPLS interfaces. This statement summary for the **encapsulation** statement describes encapsulations supported for VPLS only. For a full description of the **encapsulation-type** statement, see **encapsulation-type**.

Options

ethernet-vpls—Use Ethernet VPLS encapsulation on Ethernet interfaces that have VPLS enabled and that must accept packets carrying standard Tag Protocol ID (TPID) values. On M Series routers, except the M320 router, the 4-port Fast Ethernet TX PIC and the 1-port, 2-port, and 4-port, 4-slot Gigabit Ethernet PICs can use the Ethernet VPLS encapsulation type.

ether-vpls-over-atm-llc—For ATM intelligent queuing (IQ) interfaces only, use the Ethernet virtual private LAN service (VPLS) over ATM LLC encapsulation to bridge Ethernet interfaces and ATM interfaces over a VPLS routing instance (as described in RFC 2684, *Multiprotocol Encapsulation over ATM Adaptation Layer 5*). Packets from the ATM interfaces are converted to standard ENET2/802.3 encapsulated Ethernet frames with the frame check sequence (FCS) field removed.

extended-vlan-vpls—Use extended virtual local area network (VLAN) VPLS encapsulation on Ethernet interfaces that have VLAN 802.1Q tagging and VPLS enabled and that must accept packets carrying TPIDs 0x8100, 0x9100, and 0x9901. On M Series routers, except the M320 router, the 4-port Fast Ethernet TX PIC and the 1-port, 2-port, and 4-port, 4-slot Gigabit Ethernet PICs can use the Ethernet VPLS encapsulation type.

vlan-vpls—Use VLAN VPLS encapsulation on Ethernet interfaces with VLAN tagging and VPLS enabled. Interfaces with VLAN VPLS encapsulation accept packets carrying standard TPID values only. On M Series routers, except the M320 router, the 4-port Fast Ethernet TX PIC and the 1-port, 2-port, and 4-port, 4-slot Gigabit Ethernet PICs can use the Ethernet VPLS encapsulation type.

Required Privilege Level interface—To view this statement in the configuration.
interface-control—To add this statement to the configuration.

Related Documentation

- Configuring the VPLS Interface Encapsulation on page 485

family multiservice

Syntax

```
family multiservice {
  destination-mac;
  label-1;
  label-2;
  payload {
    ip {
      layer-3-only
    }
  }
  source-mac;
}
```

Hierarchy Level [edit forwarding-options hash-key]

Release Information Statement introduced in Junos OS Release 8.0.
label-1 statement introduced in Junos OS Release 9.4.
label-2 statement introduced in Junos OS Release 9.4.
payload statement introduced in Junos OS Release 9.4.
ip statement introduced in Junos OS Release 9.4.
layer-3-only statement introduced in Junos OS Release 9.4.

Description Configure per-packet load balancing based on the MAC addresses.

Options

destination-mac—Include the destination MAC address in the hash key used to load balance the VPLS traffic.

label-1 (M120 and M320 routers only)—Include the first MPLS label in the hash key used to load balance VPLS traffic.

label-2 (M120 and M320 routers only)—Include the second MPLS label in the hash key used to load balance VPLS traffic.

payload (M120, and M320 routers only)—Include bits from the IP payload in the hash key used to load balance VPLS traffic.

ip (M120, and M320 routers only)—Include the IP address of the IPv4 payload in the hash key used to load balance VPLS traffic.

layer-3-only (M120, and M320 routers only)—Include only the Layer 3 information from the packet's Ip payload in the hash key used to load balance VPLS traffic.

source-mac—Include the source MAC address in the hash key used to load balance the VPLS traffic.

Required Privilege Level routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation

- Configuring VPLS Load Balancing on page 489

fast-reroute-priority

Syntax fast-reroute-priority (high | low | medium);

Hierarchy Level [edit logical-systems *logical-system-name* routing-instances *routing-instance-name* forwarding-options],
[edit routing-instances *routing-instance-name* forwarding-options]

Release Information Statement introduced in Junos OS Release 9.5.

Description Specify the fast reroute priority for a VPLS routing instance. You can configure **high**, **medium**, or **low** fast reroute priority to prioritize specific VPLS routing instances for faster convergence and traffic restoration. Because the router repairs next hops for high-priority VPLS routing instances first, the traffic traversing a VPLS routing instance configured with **high** fast reroute priority is restored faster than the traffic for VPLS routing instances configured with **medium** or **low** fast reroute priority.

Default low

Options

high—Set the fast reroute priority for a VPLS routing instance to high. During a fast reroute event, the router repairs next hops for high-priority VPLS routing instances first.

low—Set the fast reroute priority for a VPLS routing instance to low, which is the default. During a fast reroute event, the router repairs next hops for low-priority VPLS routing instances last.

medium—Set the fast reroute priority for a VPLS routing instance to medium. During a fast reroute event, the router repairs next hops for medium-priority VPLS instances after high-priority VPLS routing instances but before low-priority VPLS routing instances.

Required Privilege Level routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation

- Configuring VPLS Fast Reroute Priority on page 491

interface

Syntax

```
interface interface-name {
    interface-mac-limit limit;
}
```

Hierarchy Level

[edit logical-systems *logical-system-name* routing-instances *routing-instance-name* protocols vpls site *site-name*],
[edit routing-instances *routing-instance-name* protocols vpls site *site-name*]

Release Information

Statement introduced before Junos OS Release 7.4.

Description

Configure the Layer 2 circuit pseudowires for a VPLS site as logical interfaces within the VPLS site configuration.

Options

interface-name—Specify the name of the interface used by the VPLS site.

The other option is explained separately.

Required Privilege Level

routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation

- Configuring the VPLS Site Interfaces on page 477

interface-mac-limit

Syntax

```
interface-mac-limit limit;
```

Hierarchy Level

[edit logical-systems *logical-system-name* routing-instances *routing-instance-name* protocols vpls site *site-name* interfaces *interface-name*],
[edit routing-instances *routing-instance-name* protocols vpls site *site-name* interfaces *interface-name*]

Release Information

Statement introduced before Junos OS Release 7.4.

Description

Specify the maximum number of media access control (MAC) addresses that can be learned by the VPLS routing instance. You can configure the same limit for all interfaces configured for a routing instance. You can also configure a limit for a specific interface.

Options

limit—Specify the number of MAC addresses that can be learned from each interface.

Range: 16 through 65,536 MAC addresses

Default: 512 addresses

Required Privilege Level

routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation

- Limiting the Number of MAC Addresses Learned from an Interface on page 481
- **mac-table-size on page 539**

label-block-size

Syntax label-block-size *size*;

Hierarchy Level [edit logical-systems *profile-name* routing-instances *instance-name* protocols vpls], [edit routing-instances *instance-name* protocols vpls]

Release Information Statement introduced in Junos OS Release 10.0.

Description Configure the label block size for VPLS labels.

Default 8

Options

- 2—Allocate the label blocks in increments of 2. For a VPLS domain that has only two sites with no future expansion plans.
- 4—Allocate the label blocks in increments of 4.
- 8 (default)—Allocate the label blocks in increments of 8.
- 16—Allocate the label blocks in increments of 16. A label block size of 16 enables you to minimize the number of routes in the VPLS domain. Use this setting only if the number of routes is the most important concern.

Required Privilege Level routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation

- Configuring the Label Block Size on page 521

label-switched-path-template

Syntax
```
label-switched-path-template {
    (default-template | lsp-template-name);
}
```

Hierarchy Level
```
[edit logical-systems logical-system-name routing-instances routing-instance-name
    provider-tunnel],
[edit routing-instances routing-instance-name provider-tunnel]
```

Release Information Statement introduced in Junos OS Release 8.3.

Description Enables dynamic point-to-multipoint LSPs to be used for flooding VPLS traffic. There is no default setting for the **label-switched-path-template** statement, so you must configure either the default template using the **default-template** option or you must specify the name of your preconfigured point-to-multipoint LSP template.

Options **default-template**—Create a point-to-multipoint LSP with the default parameters.

p2mp-lsp-template-name—Name of the point-to-multipoint LSP template.

Required Privilege Level routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation
- Configuring Dynamic Point-to-Multipoint Flooding LSPs on page 509

local-switching

Syntax
```
local-switching;
```

Hierarchy Level
```
[edit logical-systems logical-system-name routing-instances routing-instance-name protocols
    vpls mesh-group mesh-group-name],
[edit routing-instances routing-instance-name protocols vpls mesh-group mesh-group-name]
```

Release Information Statement introduced in Junos OS Release 9.3.

Description Allows you to terminate multiple Layer 2 circuit pseudowires at a single VPLS mesh group.

Required Privilege Level routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation
- Configuring Switching Between Pseudowires Using VPLS Mesh Groups on page 513

mac-flush

Syntax mac-flush [*explicit-mac-flush-message-options*];

Hierarchy Level [edit logical-systems *logical-system-name* routing-instances *routing-instance-name* protocols vpls],
[edit logical-systems *logical-system-name* routing-instances *routing-instance-name* protocols vpls mesh-group *mesh-group-name*],
[edit routing-instances *routing-instance-name* protocols vpls],
[edit routing-instances *routing-instance-name* protocols vpls mesh-group *mesh-group-name*]

Release Information Statement introduced in Junos OS Release 10.0.

Description Enable media access control (MAC) flush processing for the virtual private LAN service (VPLS) routing instance or for the mesh group under a VPLS routing instance. MAC flush processing removes MAC addresses from the MAC address database that have been learned dynamically. With the dynamically learned MAC addresses removed, MAC address convergence requires less time to complete.

For certain cases where MAC flush processing is not initiated by default, you can also specify ***explicit-mac-flush-message-options*** that additionally configure the router to send explicit MAC flush messages. To configure the router to send explicit MAC flush messages under specific conditions, include ***explicit-mac-flush-message-options*** with the statement.

Options *explicit-mac-flush-message-options*—(Optional) You can specify one or more of the following explicit MAC flush message options:

- **any-interface**—(Optional) Send a MAC flush message when any customer-facing attachment circuit interface goes down.
- **any-spoke**—(Optional) Send a MAC FLUSH-FROM-ME flush message to all provider edge (PE) routers in the core when one of the spoke pseudowires between the multitenant unit switch and the other network-facing provider edge (NPE) router goes down, causing the multitenant unit switch to switch to this NPE router.

  NOTE: This option has a similar effect in a VPLS multihoming environment with multiple multitenant unit switches connected to NPE routers, where both multitenant unit switches have pseudowires that terminate in a mesh group with local-switching configured. If the any-spoke option is enabled, then both PE routers send MAC FLUSH-FROM-ME flush messages to all PEs in the core.

- **propagate**—(Optional) Propagate MAC flush to the core.

Required Privilege Level routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation
- Configuring VPLS Routing Instances on page 473
- Configuring Interoperability Between BGP Signaling and LDP Signaling in VPLS on page 511

mac-table-aging-time

Syntax mac-table-aging-time *time*;

Hierarchy Level [edit logical-systems *logical-system-name* routing-instances *routing-instance-name* protocols vpls],
[edit routing-instances *routing-instance-name* protocols vpls]

Release Information Statement introduced in Junos OS Release 7.4.

Description Modify the timeout interval for the VPLS table.

Options *time*—Specify the number of seconds to wait between VPLS table clearings.

Range: 10 through 1,000,000 seconds

Default: 300 seconds

Required Privilege Level routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation
- Configuring the VPLS MAC Table Timeout Interval on page 480

mac-table-size

Syntax mac-table-size *size*;

Hierarchy Level [edit logical-systems *logical-system-name* routing-instances *routing-instance-name* protocols vpls],
[edit routing-instances *routing-instance-name* protocols vpls]

Release Information Statement introduced before Junos OS Release 7.4.

Description Modify the size of the VPLS MAC address table.

Options *size*—Specify the size of the MAC address table.

Range: (M Series and T Series) 16 through 65,536 MAC addresses
(MX Series) 16 through 1,048,575 MAC addresses

Default: 512 MAC addresses

Required Privilege Level routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation
- Configuring the Size of the VPLS MAC Address Table on page 481

mesh-group

Syntax

```
mesh-group mesh-group-name {
  local-switching;
  mac-flush [ explicit-mac-flush-message-options ];
  neighbor address {...}
  peer-as all;
  pseudowire-status-tlv;
  vpls-id number;
}
```

Hierarchy Level

[edit logical-systems *logical-system-name* routing-instances *routing-instance-name* protocols vpls],
[edit routing-instances *routing-instance-name* protocols vpls]

Release Information

Statement introduced in Junos OS Release 9.0. The **local-switching**, **mac-tlv-receive**, **mac-tlv-send**, and **peer-as** options were added in Junos OS Release 9.3. The **pseudowire-status-tlv** option was added in Junos OS Release 10.0. The **mac-flush** option was added in Junos OS Release 10.0.

Description

Specify the virtual private LAN service (VPLS) mesh group. The statement options allow you to specify each provider edge (PE) router that is a member of the mesh group. This statement is also used in the configuration of inter-autonomous system (AS) VPLS with media access control (MAC) operations.

Options

mesh-group-name—Specify the name of the VPLS mesh group.

vpls-id *number*—Specify a VPLS identifier for the mesh group.

The other statements are explained separately.

Required Privilege Level

routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation

- Configuring VPLS Routing Instances on page 473
- Configuring Interoperability Between BGP Signaling and LDP Signaling in VPLS on page 511
- Configuring Inter-AS VPLS with MAC Processing at the ASBR on page 514

multi-homing

Syntax multi-homing;

Hierarchy Level [edit logical-systems *logical-system-name* routing-instances *routing-instance-name* protocols vpls site *site-name*],
[edit routing-instances *routing-instance-name* protocols vpls site *site-name*]

Release Information Statement introduced in Junos OS Release 7.5.

Description Specify the PE router as being a part of a multihomed site. Include this statement on all PE routers associated with a particular site. Configuration of this statement tracks BGP peers. If no BGP peer is available, all active interfaces for a site are deactivated.

Required Privilege Level routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation
- Configuring Multihoming on the PE Router on page 506

neighbor

Syntax

```
neighbor neighbor-id {
    backup-neighbor {...}
    community community-name;
    encapsulation-type type;
    ignore-encapsulation-mismatch;
    pseudowire-status-tlv;
    psn-tunnel-endpoint address;
    switchover-delay milliseconds;
}
```

Hierarchy Level

[edit logical-systems *logical-system-name* routing-instances *routing-instance-name* protocols vpls],
[edit routing-instances *routing-instance-name* protocols vpls]

Release Information

Statement introduced in Junos OS Release 8.4. The **pseudowire-status-tlv** option was added in Junos OS Release 10.0.

Description

Specify each of the PE routers participating in the VPLS domain. Configuring this statement enables LDP for signaling VPLS.

Options

neighbor-id—Specify the neighbor identifier for each PE router participating in the VPLS domain.

The other options are explained separately.

Required Privilege Level

routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation

- Configuring LDP Signaling for VPLS on page 478

no-local-switching

Syntax

```
no-local-switching;
```

Hierarchy Level

[edit bridge-domains *bridge-domain-name*]

Release Information

Statement introduced in Junos OS Release 8.5.

Description

Prevents CE devices from communicating directly with each other. If the **no-local-switching** statement is configured, frames arriving on a CE interface are sent to a VPLS edge (VE) device or core-facing interfaces only.

Required Privilege Level

routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation

- Configuring VPLS and Integrated Routing and Bridging on page 510

no-tunnel-services

Syntax no-tunnel-services;

Hierarchy Level [edit logical-systems *logical-system-name* protocols vpls static-vpls],
[edit logical-systems *logical-system-name* routing-instances *routing-instance-name* protocols vpls],
[edit protocols vpls static-vpls],
[edit routing-instances *routing-instance-name* protocols vpls]

Release Information Statement introduced in Junos OS Release 7.6. Support for static VPLS added in Junos OS Release 10.2.

Description Configure VPLS on a router without a Tunnel Services PIC. Configuring the **no-tunnel-services** statement creates a label-switched interface (LSI) to provide VPLS functionality. An LSI MPLS label is used as the inner label for VPLS. This label maps to a VPLS routing instance. On the PE router, the LSI label is stripped and then mapped to a logical LSI interface. The Layer 2 Ethernet frame is then forwarded using the LSI interface to the correct VPLS routing instance.

Required Privilege Level routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation

- Configuring VPLS Without a Tunnel Services PIC on page 492
- Configuring Static Pseudowires for VPLS on page 483
- Configuring EXP-Based Traffic Classification for VPLS on page 484

peer-as

Syntax

```
peer-as {
    all;
}
```

Hierarchy Level

[edit logical-systems *logical-system-name* routing-instances *routing-instance-name* protocols vpls mesh-group *mesh-group-name*],
[edit routing-instances *routing-instance-name* protocols vpls mesh-group *mesh-group-name*]

Release Information

Statement introduced in Junos OS Release 9.3.

Description

Enable the autonomous system border router (ASBR) to establish a single pseudowire to each of the other ASBRs interconnected using inter-AS VPLS with MAC processing at the ASBR.

Options

all—This option is required. All peer routers, the ASBRs, are placed within the same VPLS mesh group.

Required Privilege Level

routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation

- Configuring Inter-AS VPLS with MAC Processing at the ASBR on page 514

rsvp-te

Syntax

```
rsvp-te {
  label-switched-path-template {
    (default-template | lsp-template-name);
  }
  static-lsp lsp-name;
}
```

Hierarchy Level

[edit logical-systems *logical-system-name* routing-instances *routing-instance-name* provider-tunnel],
[edit routing-instances *routing-instance-name* provider-tunnel]

Release Information

Statement introduced in Junos OS Release 8.3.

Description

Configure VPLS unknown unicast, broadcast, and multicast traffic flooding using point-to-multipoint LSPs.

Options

static-lsp *lsp-name*—Create a static point-to-multipoint LSP and automatically include all of the neighbors in the VPLS routing instance.

The remaining option is explained separately.

Required Privilege Level

routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation

- Flooding Unknown Traffic Using Point-to-Multipoint LSPs on page 507

site

Syntax

```
site site-name {
  interface interface-name {
    interface-mac-limit limit;
  }
  site-identifier identifier;
  site-preference preference-value;
}
```

Hierarchy Level

[edit logical-systems *logical-system-name* routing-instances *routing-instance-name* protocols vpls],
[edit routing-instances *routing-instance-name* protocols vpls]

Release Information

Statement introduced before Junos OS Release 7.4.

Description

Specify the site name and site identifier for a site. Allows you to configure a remote site ID for remote sites.

Options

site-name—Name of the site.

The remaining statements are explained separately.

Required Privilege Level

routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation

- Configuring the VPLS Site Name and Site Identifier on page 475

site-identifier

Syntax

site-identifier *identifier*;

Hierarchy Level

[edit logical-systems *logical-system-name* routing-instances *routing-instance-name* protocols vpls site *site-name*],
[edit routing-instances *routing-instance-name* protocols vpls site *site-name*]

Release Information

Statement introduced before Junos OS Release 7.4.

Description

Specify the numerical identifier for the local VPLS site.

Options

identifier—Specify the numerical identifier for the local VPLS site. The identifier must be an unsigned 16-bit number greater than zero.

Required Privilege Level

routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation

- Configuring the VPLS Site Name and Site Identifier on page 475

site-preference

Syntax

```
site-preference preference-value {
    backup;
    primary;
}
```

Hierarchy Level

[edit logical-systems *logical-system-name* routing-instances *routing-instance-name* protocols l2vpn site *site-name*],
[edit logical-systems *logical-system-name* routing-instances *routing-instance-name* protocols vpls site *site-name*],
[edit routing-instances *routing-instance-name* protocols l2vpn site *site-name*],
[edit routing-instances *routing-instance-name* protocols vpls site *site-name*]

Release Information

Statement introduced before Junos OS Release 7.4.

Description

Specify the preference value advertised for a particular Layer 2 VPN or VPLS site. The site preference value is encoded in the BGP local preference attribute. When a PE router receives multiple advertisements with the same VE identifier, the advertisement with the highest local preference value is preferred.

Options

preference-value—Specify the preference value advertised for a Layer 2 VPN or VPLS site.

Range: 1 through 65,535

backup—Set the preference value to 1.

primary—Set the preference value to 65,535.

Required Privilege Level

routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation

- Configuring the VPLS Site Preference on page 477

site-range

Syntax site-range *number*;

Hierarchy Level [edit logical-systems *logical-system-name* routing-instances *routing-instance-name* protocols vpls],
[edit routing-instances *routing-instance-name* protocols vpls]

Release Information Statement introduced before Junos OS Release 7.4.

Description Specify the maximum number of sites allowed for the VPLS domain. The value must be from 1 through 65,534.

Required Privilege Level routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation

- Configuring the Site Range on page 476

static

Syntax

```
static {
    incoming-label label;
    outgoing-label label;
}
```

Hierarchy Level

[edit logical-systems *logical-system-name* routing-instances *routing-instance-name* protocols vpls mesh-group *mesh-group-name* neighbor *address*],
[edit logical-systems *logical-system-name* routing-instances *routing-instance-name* protocols vpls neighbor *address*],
[edit logical-systems *logical-system-name* routing-instances *routing-instance-name* protocols vpls neighbor *address* backup-neighbor *address*],
[edit routing-instances *routing-instance-name* protocols vpls mesh-group *mesh-group-name* neighbor *address*],
[edit routing-instances *routing-instance-name* protocols vpls neighbor *address*],
[edit routing-instances *routing-instance-name* protocols vpls neighbor *address* backup-neighbor *address*]

Release Information

Statement introduced in Junos OS Release 10.2.

Description

Specifies a static pseudowire for a VPLS domain. By configuring static pseudowires for the VPLS domain, you do not need to configure the LDP or BGP protocols that would normally be used for signaling. Static pseudowires require that you configure a set of in and out labels for each pseudowire configured for the VPLS domain. You can configure both static and dynamic neighbors within the same VPLS routing instance. You can also configure a static pseudowire for a backup neighbor (if you configure the neighbor as static the backup must also be static) and for a mesh group.

Options

incoming-label *label*—You must configure an incoming label for the static pseudowire.

Range: 29,696 through 41,983 and 1,000,000 through 1,048,575

outgoing-label *label*—You must configure an outgoing label for the static pseudowire.

Range: 16 through 1,048,575

Required Privilege Level

routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation

- See Configuring Static Pseudowires for VPLS on page 483.

template

Syntax template;

Hierarchy Level [edit logical-systems *logical-system-name* protocols mpls label-switched-path *p2mp-lsp-template-name*],
[edit protocols mpls label-switched-path *p2mp-lsp-template-name*]

Release Information Statement introduced in Junos OS Release 8.3.

Description Specify a template for the dynamically generated point-to-multipoint LSPs used for VPLS flooding.

Required Privilege Level routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation
- Configuring Dynamic Point-to-Multipoint Flooding LSPs with a Preconfigured Template on page 510

traceoptions

Syntax

```
traceoptions {
    file filename <files number> <size size> <world-readable | no-world-readable>;
    flag flag <flag-modifier> <disable>;
}
```

Hierarchy Level

[edit logical-systems *logical-system-name* routing-instances *routing-instance-name* protocols vpls],
[edit routing-instances *routing-instance-name* protocols vpls]

Release Information

Statement introduced before Junos OS Release 7.4.

Description

Trace traffic flowing through a VPLS routing instance.

Options

disable—(Optional) Disable the tracing operation. You can use this option to disable a single operation when you have defined a broad group of tracing operations, such as **all**.

file *filename*—Name of the file to receive the output of the tracing operation. Enclose the name in quotation marks (" ").

files *number*—(Optional) Maximum number of trace files. When a trace file named ***trace-file*** reaches this size, it is renamed ***trace-file*.0**. When ***trace-file*** again reaches its maximum size, ***trace-file*.0** is renamed ***trace-file*.1** and ***trace-file*** is renamed ***trace-file*.0**. This renaming scheme continues until the maximum number of trace files is reached. Then the oldest trace file is overwritten.

If you specify a maximum number of files, you also must specify a maximum file size with the **size** option.

Range: 2 through 1000 files

Default: 2 files

flag *flag*—Tracing operation to perform. To specify more than one tracing operation, include multiple **flag** statements. You can specify the following tracing flags:

- **all**—All VPLS tracing options
- **connections**—VPLS connections (events and state changes)
- **error**—Error conditions
- **nlri**—VPLS advertisements received or sent by means of the BGP
- **route**—Routing information
- **topology**—VPLS topology changes caused by reconfiguration or advertisements received from other provider edge (PE) routers using BGP

flag-modifier—(Optional) Modifier for the tracing flag. You can specify the following modifiers:

- **detail**—Provide detailed trace information.

- **disable**—Disable the tracing flag.
- **receive**—Trace received packets.
- **send**—Trace sent packets.

no-world-readable—Do not allow any user to read the log file.

size *size*—(Optional) Maximum size of each trace file, in kilobytes (KB), megabytes (MB), or gigabytes (GB). When a trace file named ***trace-file*** reaches this size, it is renamed ***trace-file*.0**. When ***trace-file*** again reaches its maximum size, ***trace-file*.0** is renamed ***trace-file*.1** and ***trace-file*** is renamed ***trace-file*.0**. This renaming scheme continues until the maximum number of trace files is reached. Then the oldest trace file is overwritten.

If you specify a maximum file size, you also must specify a maximum number of trace files with the **files** option.

Syntax: ***x*k** to specify kilobytes, ***x*m** to specify megabytes, or ***x*g** to specify gigabytes

Range: 10 KB through the maximum file size supported on your system

Default: 1 MB

world-readable—Allow any user to read the log file.

Required Privilege Level

routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation

- Tracing VPLS Traffic and Operations on page 520

tunnel-services

Syntax

```
tunnel-services {
    devices device-names;
    primary primary-device-name;
}
```

Hierarchy Level

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

Release Information

Statement introduced before Junos OS Release 7.4.

Description

Specify that traffic for particular VPLS routing instances be forwarded to specific virtual tunnel (VT) interfaces, allowing you to load-balance VPLS traffic among all the available VT interfaces on the router.

Options

devices *device-names*—Specify the VT interfaces acceptable for use by the VPLS routing instance. If you do not configure this option, all VT interfaces available to the router can be used for de-encapsulating traffic for this instance.

primary *primary-device-name*—Specify the primary VT interface to be used by the VPLS routing instance. The VT interface specified is used to de-encapsulate all VPLS traffic from the MPLS core network for this routing instance. If the VT interface specified is unavailable, then one of the other acceptable VT interfaces is used for handling the VPLS traffic. If you do not configure this option, any acceptable VT interface can be used to de-encapsulate VPLS traffic from the core.

Required Privilege Level

routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation

- Specifying the VT Interfaces Used by VPLS Routing Instances on page 503

vlan-id

Syntax vlan-id *number*;

Hierarchy Level [edit interfaces *interface-name* unit *logical-unit-number*],
[edit logical-systems *logical-system-name* interfaces *interface-name* unit *logical-unit-number*]

Release Information Statement introduced before Junos OS Release 7.4.

Description For Fast Ethernet and Gigabit Ethernet interfaces only, bind an 802.1Q VLAN tag ID to a logical interface.

Options *number*—A valid VLAN identifier.

Range: For 4-port Fast Ethernet PICs configured to handle VPLS traffic, 512 through 1023. For 1-port and 10-port Gigabit Ethernet PICs configured to handle VPLS traffic, 512 through 4094.

Required Privilege Level interface—To view this statement in the configuration.
interface-control—To add this statement to the configuration.

Related Documentation
- Enabling VLAN Tagging on page 487

vlan-id-list (Interface in VPLS)

Syntax vlan-id-list [*numbers number–number*];

Hierarchy Level [edit interfaces *interface-name* unit *logical-unit-number*],
[edit logical-systems *logical-system-name* interfaces *interface-name* unit *logical-unit-number*]

Release Information Statement introduced for VPLS in Junos OS Release 10.2.

Description Configure a logical interface to forward packets and learn MAC addresses within each VPLS routing instance configured with a VLAN ID that matches a VLAN ID specified in the list. VLAN IDs can be entered individually using a space to separate each ID, entered as an inclusive list separating the starting VLAN ID and ending VLAN ID with a hyphen, or a combination of both.

Options *number number*—Individual VLAN IDs separated by a space.

number-number—Starting VLAN ID and ending VLAN ID in an inclusive range.

Range: 1 through 4095

Required Privilege Level interface—To view this statement in the configuration.
interface-control—To add this statement to the configuration.

Related Documentation
- Configuring Interfaces for VPLS Routing on page 484

vlan-tagging

Syntax vlan-tagging;

Hierarchy Level [edit interfaces *interface-name*],
[edit logical-systems *logical-system-name* interfaces *interface-name*]

Release Information Statement introduced before Junos OS Release 7.4.

Description For Fast Ethernet and Gigabit Ethernet interfaces only, enable the reception and transmission of 802.1Q VLAN-tagged frames on the interface.

Required Privilege Level interface—To view this statement in the configuration.
interface-control—To add this statement to the configuration.

Related Documentation

- Enabling VLAN Tagging on page 487

vpls

See the following sections:

- **vpls (Interfaces) on page 556**
- **vpls (Routing Instance) on page 557**

vpls (Interfaces)

Syntax vpls;

Hierarchy Level [edit interfaces *interface-name* unit *logical-unit-number* family],
[edit logical-systems *logical-system-name* interfaces *interface-name* unit *logical-unit-number* family]

Release Information Statement introduced before Junos OS Release 7.4.

Description Specify the VPLS protocol family information for the logical interface.

Required Privilege Level routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation

- Configuring Interfaces for VPLS Routing on page 484

vpls (Routing Instance)

Syntax

```
vpls {
  active-interface {
    any;
    primary interface-name;
  }
  connectivity-type (ce | irb);
  interface-mac-limit limit;
  label-block-size size;
  mac-flush [ explicit-mac-flush-message-options ];
  mac-table-aging-time time;
  mac-table-size size;
  mesh-group mesh-group-name {
    local-switching;
    mac-flush [ explicit-mac-flush-message-options ];
    neighbor address;
    peer-as all;
  vpls-id number;
  no-tunnel-services;
  site site-name {
    interface interface-name {
      interface-mac-limit limit;
    }
    multi-homing;
    site-identifier identifier;
    site-preference preference-value;
  }
  site-range number;
  traceoptions {
    file filename <files number> <size size> <world-readable | no-world-readable>;
    flag flag <flag-modifier> <disable>;
  }
  tunnel-services {
    devices device-names;
    primary primary-device-name;
  }
}
```

Hierarchy Level

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

Release Information

Statement introduced before Junos OS Release 7.4. The **mac-flush** option was added in Junos OS Release 10.0.

Description

Configure a virtual private LAN service (VPLS) routing instance.

The remaining statements are explained separately.

Required Privilege Level

routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation
- Configuring VPLS Routing Instances on page 473

vpls-id

Syntax vpls-id *vpls-id*;

Hierarchy Level [edit logical-systems *logical-system-name* routing-instances *routing-instance-name* protocols vpls],
[edit routing-instances *routing-instance-name* protocols vpls]

Release Information Statement introduced in Junos OS Release 8.4.

Description Identify the virtual circuit identifier used for the VPLS routing instance. This statement is a part of the configuration to enable LDP signaling for VPLS.

Options ***vpls-id***—Specify a valid identifier for the VPLS routing instance.

Required Privilege Level routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation
- Configuring LDP Signaling for VPLS on page 478

PART 6

Interprovider and Carrier-of-Carriers

CHAPTER 22

Introduction to Interprovider and Carrier-of-Carriers VPNs

This chapter discusses the following topics, which provide background information about carrier-of-carriers VPNs:

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.

The sections that follow provide an overview of traditional VPNs, interprovider and carrier-of-carriers VPNs, and the differences in how external and internal routes are handled in each of these environments.

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.

Standard VPNs

The traditional distinction between internal routes and external routes also applies to VPN routing architectures. As shown in Figure 1 on page 3, the provider (P) routers maintain only the service provider's internal routes (to provider edge [PE] routers and other P routers); they do not maintain VPN routes. PE routers are the only devices in the provider network that are required to maintain external routes.

The BGP next hop connects the external routes to the internal routes in traditional VPNs:

- The BGP next hop is advertised with each external route in BGP advertisements.
- The route to the BGP next hop is an internal route that is advertised by the IGP.
- MPLS provides packet forwarding from the ingress PE router to the BGP next-hop egress PE router.

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 563—The customer sites belong to different ASs. You need to configure EBGP to exchange the customer's external routes.
- "Carrier-of-Carriers VPNs" on page 565—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.

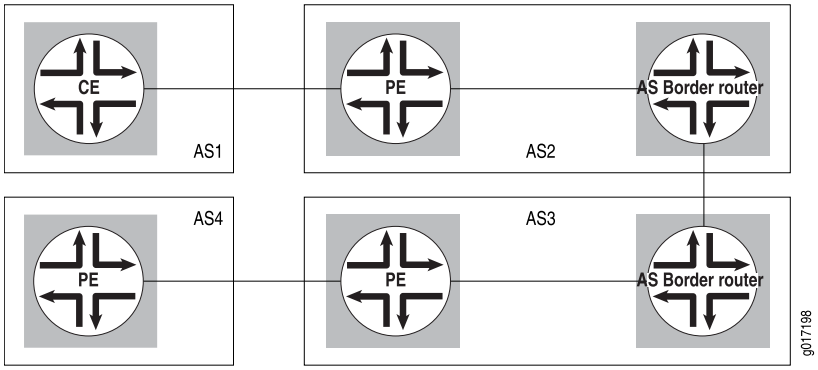
The following are definitions of the types of PE routers specific to interprovider and carrier-of-carriers VPNs:

- 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 VPNs

Interprovider VPNs provide connectivity between separate ASs. This functionality might be used by a VPN customer who has connections to several different ISPs, or different connections to the same ISP in different geographic regions, each of which has a different AS. Figure 59 on page 563 illustrates the type of network topology used by an interprovider VPN.

Figure 59: Interprovider VPN Network Topology



The following sections describe the ways you can configure an interprovider VPN:

- Linking VRF Tables Between Autonomous Systems on page 563
- Configuring MP-EBGP Between AS Border Routers on page 563
- Configuring Multihop MP-EBGP Between AS Border Routers on page 564

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 of one AS to the VRF table in the AS border router in the other AS. Each AS border router must contain a VRF instance for every VPN configured in both service provider networks. You then configure an IP session between the two AS border routers. In effect, the AS border routers 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 in this manual.

Configuring MP-EBGP Between AS Border Routers

In this approach, the PE routers within an AS use multiprotocol external BGP (MP-EBGP) to distribute labeled VPN–Internet Protocol version 4 (IPv4) routes to an AS border router or to a route reflector of which the AS border router is a client. The AS border router uses multiprotocol external BGP (MP-EBGP) to distribute the labeled VPN-IPv4 routes to its peer AS border router in the neighboring AS. The peer AS border router 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.

This approach enhances the scalability of an EBGP VRF-to-VRF configuration because it eliminates the need to configure all the VPNs on every AS border router. However, it also introduces some complexity:

- All the VRF routes must be stored in the AS border router.
- An LSP must be established from ingress PE routers to egress PE routers.
- Secure connections must exist among the ASs along the path from the ingress PE router to the egress PE router.
- The ASs must be configured to store information about which AS border routers receive routes with specific route target attributes.

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.

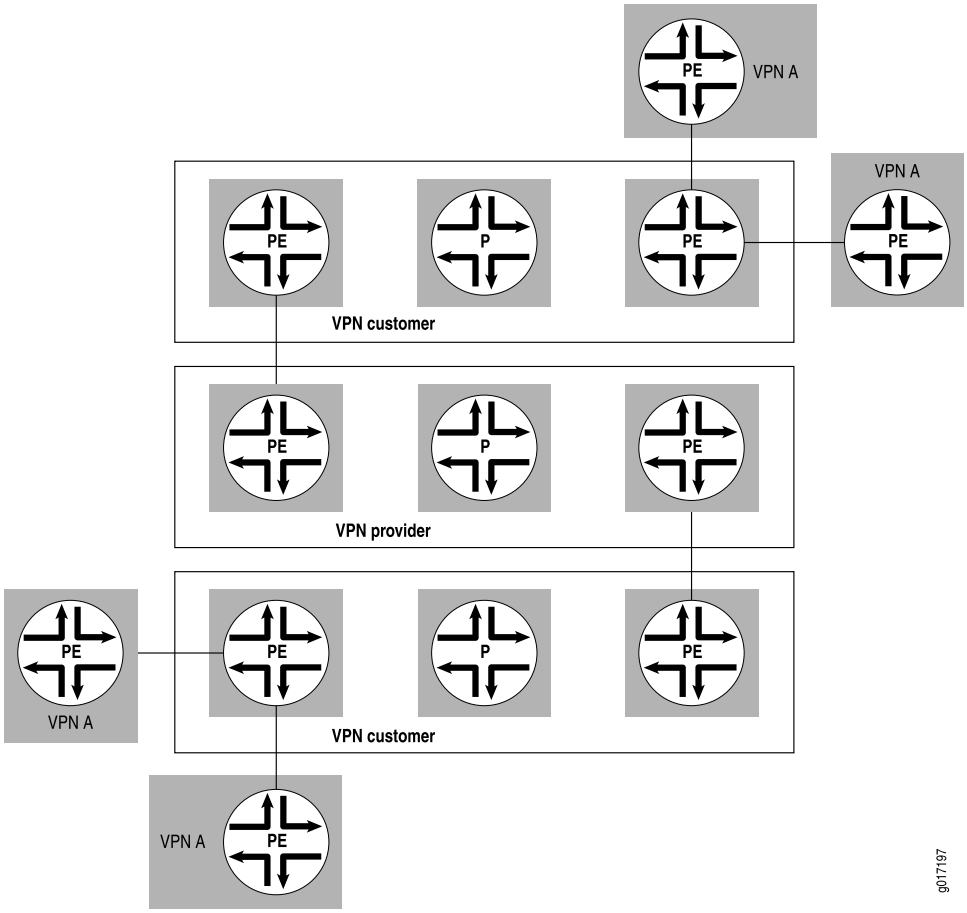
Carrier-of-Carriers VPNs

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 566—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 566—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 60 on page 565 illustrates the network architecture used for a carrier-of-carriers VPN service.

Figure 60: 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 13 on page 566.

Table 13: 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

Interprovider and Carrier-of-Carriers VPN Standards

Interprovider and carrier-of-carriers VPNs are defined by the following documents:

- RFC 3107, *Carrying Label Information in BGP-4*
- RFC 4364, *BGP/MPLS IP Virtual Private Networks (VPNs)*

To access Internet RFCs and drafts, go to the IETF website at **http://www.ietf.org/**.

CHAPTER 23

Configuring Interprovider and Carrier-of-Carriers VPNs

This chapter describes how to configure interprovider and carrier-of-carriers VPNs, discussing the following topics:

Configuring Interprovider VPNs

You can configure interprovider VPN service using either multiprotocol external BGP (MP-EBGP) or multihop MP-EBGP:

Configuring Interprovider VPNs Using MP-EBGP

To configure interprovider VPN service using MP-EBGP, you need to configure the AS border routers of each AS. For an illustration of how the routers interconnect in an interprovider VPN service, see Figure 59 on page 563.

The configuration of the AS border routers in each AS is nearly identical. To configure each AS border router, you perform the steps in the following sections:

Configuring RSVP

You need to configure the interprovider VPN interface in RSVP. This interface on the PE router, which handles VPN traffic in the current AS, receives VPN traffic from the other AS.

Configure the interface for RSVP by including the **interface** statement:

```
interface interface-name;
```

You can include this statement at the following hierarchy levels:

- **[edit protocols rsvp]**
- **[edit logical-systems *logical-system-name* protocols rsvp]**

Configuring MPLS

Configure a label-switched path (LSP) to the PE router. Also configure the interfaces handling VPN traffic from the other AS and to the PE router in the current AS.

```
mpls {
    label-switched-path path-name {
        to address;
    }
    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

Configure an MP-EBGP session on the AS border router. This session exchanges VPN Internet Protocol version 4 (IPv4) routes with the AS border router in the other AS.

To configure a group to handle IBGP and a group to handle EBGP, include the **bgp** statement:

```
bgp {
    keep all;
    group group-name {
        type internal;
        local-address address;
        family inet-vpn {
            unicast;
        }
        neighbor address;
    }
    group group-name {
        type external;
        family inet-vpn {
            unicast;
```

```
        }
        neighbor address {
          peer-as as number;
        }
      }
    }
```

You can include this statement at the following hierarchy levels:

- **[edit protocols]**
- **[edit logical-systems *logical-system-name* protocols]**

Configuring OSPF

To configure OSPF on the AS border router, include the **ospf** statement:

```
ospf {
  traffic engineering;
  area address {
    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 Interprovider VPNs Using Multihop MP-EBGP

To configure a network to provide interprovider VPN service using multihop MP-EBGP, you need to set up the AS border routers and the PE routers connected to the end customer's CE routers. For an illustration of how the routers interconnect in an interprovider VPN service, see Figure 59 on page 563.

The following sections describe how to configure a network to provide interprovider VPN service using multihop MP-EBGP:

- Configuring the AS Border Routers on page 571
- Configuring the PE Router on page 573

Configuring the AS Border Routers

The configuration of the AS border routers in each AS is nearly identical. To configure each AS border router, you perform the steps in the following sections:

- Configuring BGP on page 572
- Configuring Policy Options on page 572

Configuring BGP

Configure BGP on the AS border routers. To configure a group for IBGP to the PE router, include the **bgp** statement:

```
bgp {
  group group-name {
    type internal;
    local-address address;
    family inet {
      labeled-unicast {
        resolve-vpn;
      }
    }
    neighbor address;
  }
}
```

To configure a group for EBGP to the AS border router in the adjacent AS router, include the **bgp** statement:

```
bgp {
  group group-name {
    type external;
    family inet {
      labeled-unicast;
    }
    export internal;
    neighbor address {
      peer-as as-number;
    }
  }
}
```

You can include this statement at the following hierarchy levels:

- **[edit protocols]**
- **[edit logical-systems** *logical-system-name* **protocols]**

Configuring Policy Options

For the policy configuration on the AS border routers, you only need to advertise the loopbacks of the PE routers. If the AS border router is also a PE router, configure **from protocol ospf direct** at the **[edit policy-options policy-statement** *policy-name* **term** *term-name***]** hierarchy level.

To configure the policy options on the AS border routers, include the **policy-statement** statement:

```
policy-statement policy-name {
  term term-name {
    from {
      protocol ospf direct;
      route-filter pe-router-loopback-address exact accept;
    }
    then reject;
```

```
    }
  }
```

You can include these statements at the following hierarchy levels:

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

Configuring the PE Router

Configure a multihop MP-EBGP session on the PE router connected to the end customer's CE router.

To pass labeled IPv4 routes, include the **labeled-unicast** statement:

```
labeled-unicast {
  resolve-vpn;
}
```

You can include this statement at the following hierarchy levels:

- **[edit protocols bgp group** *group-name* **family inet]**
- **[edit logical-systems** *logical-system-name* **protocols bgp group** *group-name* **family inet]**

To configure a group to handle an EBGP multihop session with the remote PE router (that is, to pass VPN-IPv4 routes), include the **bgp** statement:

```
bgp {
  group group-name {
    multihop {
      ttl 10;
    }
    family inet-vpn {
      unicast;
    }
  }
  neighbor address {
    peer-as as-number;
  }
}
```

You can include this statement at the following hierarchy levels:

- **[edit protocols]**
- **[edit logical-systems** *logical-system-name* **protocols]**

Configuring Carrier-of-Carriers VPNs for Customers That Provide Internet Service

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. Figure 60 on page 565 shows how the routers 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 acts as a CE router with respect to the service provider's PE router. The following sections describe how to configure the carrier-of-carriers VPN service customer's CE router:

Configuring MPLS

To configure MPLS on the customer's CE router, 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 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, 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, 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, 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, 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, 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, include the **labeled-unicast** statement in the configuration for the routing instance so the PE router can send labels to the customer's CE router:

```
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, 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, 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]**

Configuring Carrier-of-Carriers VPNs for Customers That Provide VPN Service

You can configure a carrier-of-carriers VPN service for customers who want VPN service. Figure 60 on page 565 shows how the routers in this type of service interconnect.

To configure the following routers in the customer's and provider's networks to enable carrier-of-carriers VPN service, you perform the steps in the following sections:

- Configuring the Carrier-of-Carriers Customer's PE Router on page 579
- Configuring the Carrier-of-Carriers Customer's CE Router on page 582
- Configuring the Provider's PE Router on page 584

Configuring the Carrier-of-Carriers Customer's PE Router

The carrier-of-carriers customer's PE router is connected to the end customer's CE router.

The following sections describe how to configure the carrier-of-carriers customer's PE router:

- Configuring MPLS on page 579
- Configuring BGP on page 579
- Configuring OSPF on page 580
- Configuring LDP on page 580
- Configuring VPN Service in the Routing Instance on page 581
- Configuring Policy Options on page 581

Configuring MPLS

To configure MPLS on the carrier-of-carriers customer's PE router, 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 (see "Configuring Carrier-of-Carriers VPNs for Customers That Provide VPN Service" on page 579), and include the **family-inet-vpn** statement in the configuration for the IBGP session to the carrier-of-carriers PE router 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, 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, 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 on the carrier-of-carriers customer's PE router, 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, 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

The carrier-of-carriers customer's CE router connects to the provider's PE router. Complete the instructions in the following sections to configure the carrier-of-carriers customers' CE router:

- Configuring MPLS on page 582
- Configuring BGP on page 582
- Configuring OSPF and LDP on page 583
- Configuring Policy Options on page 583

Configuring MPLS

In the MPLS configuration for the carrier-of-carriers customer's CE router, include the interfaces to the provider's PE router and to a P router 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, configure a group that includes the **labeled-unicast** statement to extend VPN service to the PE router connected to the end customer's CE router:

```
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, 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, 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, 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

The carrier-of-carriers provider's PE routers connect to the carrier customer's CE routers. Complete the instructions in the following sections to configure the provider's PE router:

- Configuring MPLS on page 584
- Configuring a PE-Router-to-PE-Router BGP Session on page 584
- Configuring IS-IS and LDP on page 585
- Configuring Policy Options on page 585
- Configuring a Routing Instance to Send Routes to the CE Router on page 586

Configuring MPLS

In the MPLS configuration, specify at least two interfaces—one to the customer's CE router and one to connect to the provider's PE router 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-Router-to-PE-Router BGP Session

To configure a PE-router-to-PE-router BGP session on the provider's PE routers to allow VPN-IPv4 routes to pass between the PE routers, 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, 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 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 to send labeled routes to the carrier customer's CE router, 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***]**

Configuring BGP to Gather Interprovider and Carrier-of-Carriers VPNs Statistics

You can configure BGP to gather traffic statistics for interprovider and carrier-of-carriers VPNs.

To configure BGP to gather traffic statistics for interprovider and carrier-of-carriers VPNs, include the **traffic-statistics** statement:

```
traffic-statistics {
   file filename <world-readable | no-world-readable>;
   interval seconds;
}
```

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

NOTE: Traffic statistics for interprovider and carrier-of-carriers VPNs are available only for IPv4. IPv6 is not supported.

If you do not specify a filename, the statistics are not written to a file. However, if you have included the **traffic-statistics** statement in the BGP configuration, the statistics are still available and can be accessed by means of the **show bgp group traffic-statistics** ***group-name*** command.

To account for traffic from each customer separately, separate labels must be advertised for the same prefix to the peer routers in different groups. To enable separate traffic accounting, you need to include the **per-group-label** statement in the configuration for each BGP group. By including this statement, statistics are collected and displayed that account for traffic sent by the peers of the specified BGP group.

If you configure the statement at the **[edit protocols bgp family inet]** hierarchy level, rather than configuring it for a specific BGP group, then the traffic statistics are shared with all BGP groups configured with the **traffic-statistics** statement but not configured with the **per-group-label** statement.

To account for traffic from each customer separately, include the **per-group-label** statement in the configuration for each BGP group:

per-group-label;

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

The following shows a sample of the output to the traffic statistics file:

```
Dec 19 10:39:54 Statistics for BGP group ext2 (Index 1) NLRI inet-labeled-unicast
Dec 19 10:39:54  FEC               Packets         Bytes   EgressAS   FECLabel
Dec 19 10:39:54 10.255.245.55            0             0          I     100160
Dec 19 10:39:54 10.255.245.57            0             0          I     100112
Dec 19 10:39:54 100.101.0.0              0             0         25     100080
Dec 19 10:39:54 100.102.0.0              0             0         25     100080
Dec 19 10:39:54 100.103.0.0            109          9592         25     100048
Dec 19 10:39:54 100.104.0.0            109          9592         25     100048
Dec 19 10:39:54 192.168.25.0             0             0          I     100064
Dec 19 10:39:54  Dec 19 10:39:54, read statistics for 5 FECs in 00:00:00 seconds
 (10 queries) for BGP group ext2 (Index 1) NLRI inet-labeled-unicast
```

CHAPTER 24

Configuration Examples for Interprovider and Carrier-of-Carriers VPNs

This chapter contains examples that illustrate how to configure interprovider and carrier-of-carriers virtual private networks (VPNs). It includes the following sections:

Example Terminology

B

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.

M

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.

R

***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.

V

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.

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.

Interprovider VPN Example—MP-EBGP Between ISP Peer Routers

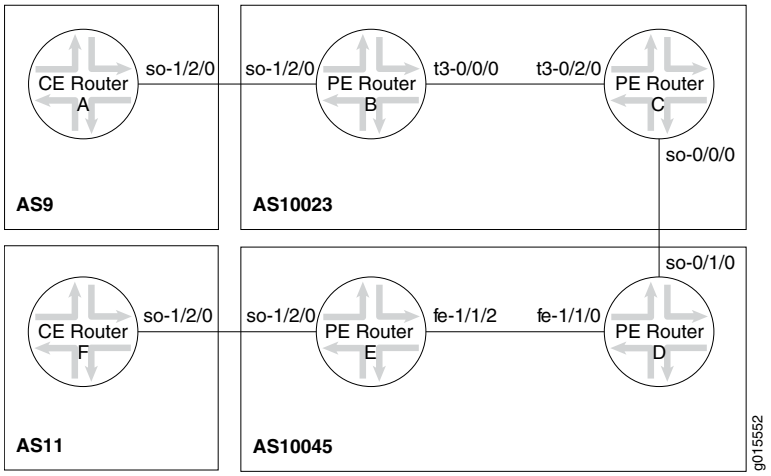
In this example, all routes learned from the CE routers are sent over both service provider networks as VPN-IPv4 routes. The routes are initially learned by the PE routers (Router B and Router E) from the CE routers (Router A and Router F) and are announced by the PE routers to the AS border routers (Router C and Router D). The AS border routers are then configured with an MP-EBGP session, enabling them to pass the VPN-IPv4 routes with each other. When an AS border router—Router C for example—learns VPN-IPv4 routes from an IBGP PE, the following events occur:

1. Router C sets itself as the next hop for the route and creates a label for that route.
2. Router C advertises the VPN-IPv4 route to PE Router D in AS 10045.
3. Router D sets the next hop to itself, creates another label, and then forwards the label and the route to its IBGP PE router (Router E).

This example has scaling limitations because of restrictions on the number of labels each PE router needs to allocate at the AS border.

Figure 61 on page 591 illustrates the network topology used in this VPN example.

Figure 61: Network Topology for the Interprovider VPN Example



For configuration information see the following sections:

- Configuration for Router A on page 591
- Configuration for Router B on page 591
- Configuration for Router C on page 593
- Configuration for Router D on page 594
- Configuration for Router E on page 595
- Configuration for Router F on page 596

Configuration for Router A

Configure a family **inet** EBGP session with Router B and export the direct routes:

```
[edit]
protocols {
   bgp {
      group to-provider {
         export attached;
         peer-as 10023;
         neighbor 192.168.198.2;
      }
   }
}
policy-options {
   policy-statement attached {
      from protocol direct;
      then accept;
   }
}
```

Configuration for Router B

Router A is configured as a CE router (using the **routing-instances** statement) in the configuration for Router B. Because they exchange VPN-IPv4 routes, Router D and Router C are configured as PE routers.

Configure Router B:

```
[edit]
protocols {
    rsvp {
        interface t3-0/0/0.0;
    }
    mpls {
        label-switched-path to-routerC {
            to 10.255.14.171;
            description "to-routerC for use with VPNs";
        }
        interface t3-0/0/0.0;
        interface so-1/2/0.0;
    }
    bgp {
        group to-ibgp {
            type internal;
            local-address 10.255.14.175;
            family inet-vpn {
                unicast;
            }
            neighbor 10.255.14.171;
        }
    }
    ospf {
        traffic-engineering;
        reference-bandwidth 4g;
        area 0.0.0.0 {
            interface t3-0/0/0.0;
            interface lo0.0 {
                passive;
            }
        }
    }
}
routing-instances {
    vpna {
        instance-type vrf;
        interface so-1/2/0.0;
        route-distinguisher 10.255.14.175:9;
        vrf-import vpna-import;
        vrf-export vpna-export;
        protocols {
            bgp {
                group to-ce {
                    peer-as 9;
                    neighbor 192.168.198.1;
                }
            }
        }
    }
}
policy-options {
    policy-statement vpna-import {
        term 1 {
```

```
        from {
          protocol bgp;
          community vpna-comm;
        }
        then accept;
      }
      term 2 {
        then reject;
      }
    }
    policy-statement vpna-export {
      term 1 {
        from protocol bgp;
        then {
          community add vpna-comm;
          accept;
        }
      }
      term 2 {
        then reject;
      }
    }
    community vpna-comm members target:100:1001;
}
```

Configuration for Router C

In the BGP protocol configuration for Router C, include the **keep all** statement. When this statement is included, BGP must store every route learned through BGP. Configure two BGP sessions (configure **family inet-vpn** on both sessions):

- IBGP session to Router B (group **to-ibgp** in this example)
- EBGP session to Router D (group **to-ebgp-pe** in this example)

Interface **t3-0/2/0** is added at the **[edit protocols mpls]** hierarchy level, allowing BGP to announce routes with labels over the EBGP session.

Configure Router C:

```
[edit]
protocols {
  rsvp {
    interface t3-0/2/0.0;
  }
  mpls {
    label-switched-path to-routerB {
      to 10.255.14.175;
      description "to-routerB for use with vpns";
    }
    interface t3-0/2/0.0;
    interface so-0/0/0.0;
  }
  bgp {
    keep all;
    group to-ibgp {
```

```
        type internal;
        local-address 10.255.14.171;
        family inet-vpn {
          unicast;
        }
        neighbor 10.255.14.175;
      }
      group to-ebgp-pe {
        type external;
        family inet-vpn {
          unicast;
        }
        neighbor 192.168.197.22 {
          peer-as 10045;
        }
      }
    }
    ospf {
      traffic-engineering;
      reference-bandwidth 4g;
      area 0.0.0.0 {
        interface t3-0/2/0.0;
        interface lo0.0 {
          passive;
        }
      }
    }
}
```

Configuration for Router D

The configuration for Router D is almost identical to that of Router C:

```
[edit]
protocols {
  rsvp {
    interface fe-1/1/0.0;
  }
  mpls {
    label-switched-path to-E {
      to 10.255.14.177;
      description "to-routerE for vpna";
    }
    interface fe-1/1/0.0;
    interface so-0/1/0.0;
  }
  bgp {
    keep all;
    group to-ibgp-pe {
      type internal;
      family inet-vpn {
        unicast;
      }
      neighbor 10.255.14.177;
    }
    group to-ebgp-pe {
```

```
        type external;
        family inet-vpn {
          unicast;
        }
        peer-as 10023;
        neighbor 192.168.197.21;
      }
    }
    ospf {
      traffic-engineering;
      reference-bandwidth 4g;
      area 0.0.0.0 {
        interface fe-1/1/0.0;
        interface lo0.0 {
          passive;
        }
      }
    }
}
```

Configuration for Router E

The configuration for Router E is very similar to the configuration for Router B:

```
[edit]
protocols {
  rsvp {
    interface fe-1/1/2.0;
  }
  mpls {
    label-switched-path to-routerD {
      to 10.255.14.173;
      description "to-routerD for use with VPNa";
    }
    interface fe-1/1/2.0;
    interface so-1/2/0.0;
  }
  bgp {
    group to-ibgp-pe {
      type internal;
      local-address 10.255.14.177;
      family inet-vpn {
        unicast;
      }
      neighbor 10.255.14.173;
    }
  }
  ospf {
    traffic-engineering;
    reference-bandwidth 4g;
    area 0.0.0.0 {
      interface fe-1/1/2.0;
      interface lo0.0 {
        passive;
      }
    }
```

```
    }
  }
  routing-instances {
    vpna {
      instance-type vrf;
      interface so-1/2/0.0;
      route-distinguisher 10.255.14.177:11;
      vrf-import vpna-import;
      vrf-export vpna-export;
      protocols {
        bgp {
          group to-routerF-ce {
            neighbor 192.168.198.14 {
              peer-as 11;
            }
          }
        }
      }
    }
  }
  policy-options {
    policy-statement vpna-import {
      term 1 {
        from {
          protocol bgp;
          community vpna-comm;
        }
        then accept;
      }
      term 2 {
        then reject;
      }
    }
    policy-statement vpna-export {
      term 1 {
        from protocol bgp;
        then {
          community add vpna-comm;
          accept;
        }
      }
      term 2 {
        then reject;
      }
    }
    community vpna-comm members target:100:1001;
  }
```

Configuration for Router F

Configure Router F as a CE router; the configuration is similar to that for Router A:

```
[edit]
protocols {
  bgp {
    group to-provider {
```

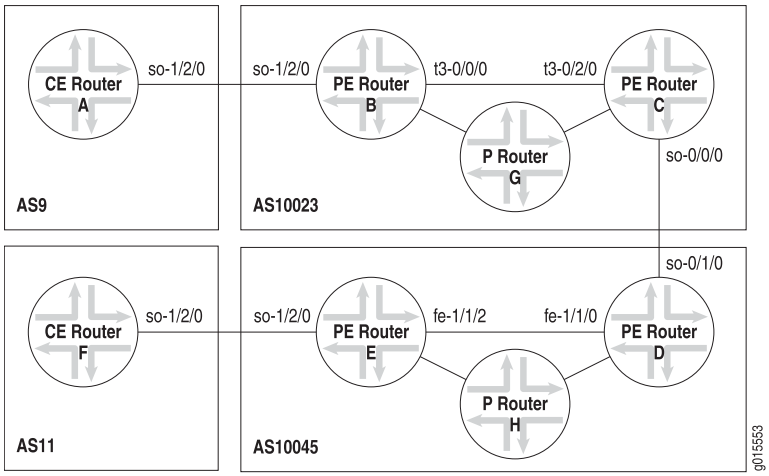
```
            type external;
            export attached;
            neighbor 192.168.198.13 {
              peer-as 10045;
            }
          }
        }
      }
      policy-options {
        policy-statement attached {
          from protocol direct;
          then accept;
        }
      }

}
```

Interprovider VPN Example—Multihop MP-EBGP with P Routers

In this example, labeled IPv4 (not VPN-IPv4), routes are exchanged by the AS border routers (Router C and Router D) to provide MPLS connectivity between the PE routers. Router G and H are provider routers.

Figure 62 on page 597 illustrates the network topology used in this VPN example.

Figure 62: Network Topology of Interprovider VPN Example—Multihop MP-EBGP



Only routes internal to the service provider networks should be announced between Router C and Router D. Configure this by including the **family inet labeled-unicast** statement in the IBGP and EBGP configuration on the PE routers. When you set **family inet labeled-unicast**, the local router announces internal routes from **inet.0** in the following manner:

- If a label exists for the route, the local router creates a label, performs a swap, and announces the route from **inet.0** with the label.

- If a label does not exist for the route, the local router creates a label, performs a pop, and announces the route from **inet.0** with the label.

Routes learned from the **labeled-unicast** session are placed into the **inet.0** routing table.

In addition, you configure a multihop MP-EBGP session between the end PE routers (Router B and Router E). This additional MP-EBGP session allows the announcement of VPN-IPv4 routes, and allows you to maintain VPN connectivity while keeping VPN-IPv4 routes out of the core of the network.

For configuration information, see the following sections:

- Configuration for Router A on page 598
- Configuration for Router B on page 598
- Configuration for Router C on page 600
- Configuration for Router D on page 601
- Configuration for Router E on page 602
- Configuration for Router F on page 604

Configuration for Router A

The configuration for Router A in this example is identical to the configuration for Router A in "Interprovider VPN Example—MP-EBGP Between ISP Peer Routers" on page 590. See "Configuration for Router A" on page 591

Configuration for Router B

Router A is configured as a CE router (using the **routing-instances** statement) in the configuration for Router B. Because they exchange VPN-IPv4 routes, Router C and Router D are configured as PE routers.

In the BGP group **to-ibgp**, include the **family inet labeled-unicast** statement to pass labeled IPv4 routes, and configure an EBGP multihop session to pass VPN-IPv4 routes:

```
[edit]
protocols {
   bgp {
      group to-ibgp {
         type internal;
         local-address 10.255.14.175;
         family inet {
            labeled-unicast {
               resolve-vpn;
            }
         }
         neighbor 10.255.14.171;
      }
      group to-remote-pe {
         multihop {
            ttl 10;
         }
         family inet-vpn {
            unicast;
```

```
    }
    neighbor 10.255.14.177 {
      peer-as 10045;
    }
  }
  mpls {
    label-switched-path to-routerC {
      to 10.255.14.171;
      description "to-routerC for use with VPNs";
    }
    interface t3-0/0/0.0;
    interface so-1/2/0.0;
  }
  ospf {
    traffic-engineering;
    reference-bandwidth 4g;
    area 0.0.0.0 {
      interface t3-0/0/0.0;
      interface lo0.0 {
        passive;
      }
    }
  }
  rsvp {
    interface t3-0/0/0.0;
  }
}
routing-instances {
  vpna {
    instance-type vrf;
    interface so-1/2/0.0;
    route-distinguisher 10.255.14.175:9;
    vrf-import vpna-import;
    vrf-export vpna-export;
    protocols {
      bgp {
        group to-ce {
          peer-as 9;
          neighbor 192.168.198.1;
        }
      }
    }
  }
}
policy-options {
  policy-statement vpna-import {
    term 1 {
      from {
        protocol bgp;
        community vpna-comm;
      }
      then accept;
    }
    term 2 {
      then reject;
    }
```

```
    }
    policy-statement vpna-export {
      term 1 {
        from protocol bgp;
        then {
          community add vpna-comm;
          accept;
        }
      }
      term 2 {
        then reject;
      }
    }
    community vpna-comm members target:100:1001;
  }
}
```

Configuration for Router C

Configure two BGP sessions (configure **family inet-vpn** on both sessions):

- IBGP session to Router B (group **to-ibgp** in this example)
- EBGP session to Router D (group **to-ebgp-pe** in this example)

Interface **t3-0/2/0** is added at the **[edit protocols mpls]** hierarchy level, allowing BGP to announce routes with labels over the EBGP session.

Configure Router C:

```
[edit]
protocols {
  bgp {
    group to-ibgp {
      type internal;
      local-address 10.255.14.171;
      family inet {
        labeled-unicast;
      }
      neighbor 10.255.14.175;
    }
    group to-ebgp-pe {
      type external;
      family inet {
        labeled-unicast;
      }
      export internal;
      neighbor 192.168.197.22 {
        peer-as 10045;
      }
    }
    mpls {
      label-switched-path to-routerB {
        to 10.255.14.175;
        description "to-routerB for use with vpns";
      }
```

```
        interface t3-0/2/0.0;
        interface so-0/0/0.0;
        traffic-engineering bgp-igp;
      }
      rsvp {
        interface t3-0/2/0.0;
      }
      ospf {
        traffic-engineering;
        reference-bandwidth 4g;
        area 0.0.0.0 {
          interface t3-0/2/0.0;
          interface lo0.0 {
            passive;
          }
        }
      }
    }
    policy-options {
      policy-statement internal {
        term 1 {
          from protocol [ospf direct ldp];
          then accept;
        }
        term 2 {
          then reject;
        }
      }
    }
}
```

Configuration for Router D

Configure Router D:

```
[edit]
protocols {
  bgp {
    group to-ibgp-pe {
      type internal;
      family inet {
        labeled-unicast;
      }
      neighbor 10.255.14.177;
    }
    group to-ebgp-pe {
      type external;
      family inet {
        labeled-unicast;
      }
      export internal;
      peer-as 10023;
      neighbor 192.168.197.21;
    }
    mpls {
      label-switched-path to-E {
```

```
            to 10.255.14.177;
            description "to-routerE for vpna";
          }
          interface fe-1/1/0.0;
          interface so-0/1/0.0;
          traffic-engineering bgp-igp;
        }
        ospf {
          traffic-engineering;
          reference-bandwidth 4g;
          area 0.0.0.0 {
            interface fe-1/1/0.0;
            interface lo0.0 {
              passive;
            }
          }
        }
        rsvp {
          interface fe-1/1/0.0;
        }
      }
      policy-options {
        policy-statement internal {
          term 1 {
            from protocol [ospf direct ldp];
            then accept;
          }
          term 2 {
            then reject;
          }
        }
      }
    }
```

Configuration for Router E

The configuration for Router E is very similar to the configuration for Router B:

```
[edit]
protocols {
  bgp {
    group to-ibgp-pe {
      type internal;
      local-address 10.255.14.177;
      family inet {
        labeled-unicast;
      }
      neighbor 10.255.14.173;
    }
    group to-remote-pe {
      multihop {
        ttl 10;
      }
      family inet-vpn {
        unicast;
      }
```

```
    neighbor 10.255.14.175 {
      peer-as 10023;
    }
  }
  mpls {
    label-switched-path to-routerD {
      to 10.255.14.173;
      description "to-routerD for use with VPNa";
    }
    interface fe-1/1/2.0;
    interface so-1/2/0.0;
  }
  ospf {
    traffic-engineering;
    reference-bandwidth 4g;
    area 0.0.0.0 {
      interface fe-1/1/2.0;
      interface lo0.0 {
        passive;
      }
    }
  }
  rsvp {
    interface fe-1/1/2.0;
  }
}
routing-instances {
  vpna {
    instance-type vrf;
    interface so-1/2/0.0;
    route-distinguisher 10.255.14.177:11;
    vrf-import vpna-import;
    vrf-export vpna-export;
    protocols {
      bgp {
        group to-routerF-ce {
          neighbor 192.168.198.14 {
            peer-as 11;
          }
        }
      }
    }
  }
}
policy-options {
  policy-statement vpna-import {
    term 1 {
      from {
        protocol bgp;
        community vpna-comm;
      }
      then accept;
    }
    term 2 {
      then reject;
    }
```

```
        }
        policy-statement vpna-export {
          term 1 {
            from protocol bgp;
            then {
              community add vpna-comm;
              accept;
            }
          }
          term 2 {
            then reject;
          }
        }
        community vpna-comm members target:100:1001;
      }
    }
```

Configuration for Router F

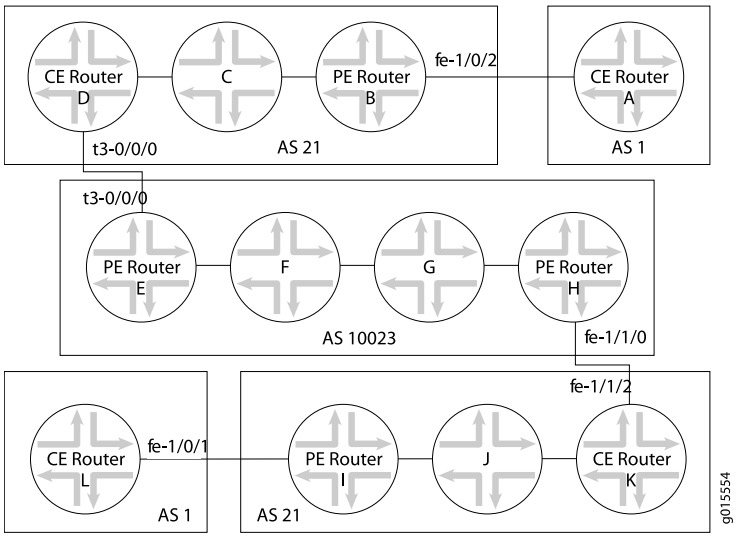
The configuration for Router F in this example is identical to the configuration for Router F in "Interprovider VPN Example—MP-EBGP Between ISP Peer Routers" on page 590. See "Configuration for Router F" on page 596.

Carrier-of-Carriers VPN Examples

A carrier-of-carriers service allows an Internet service provider (ISP) to connect to a transparent outsourced backbone at multiple locations.

Figure 63 on page 604 shows the network topology in both carrier-of-carriers examples.

Figure 63: Carrier-of-Carriers VPN Example Network Topology



Carrier-of-Carriers VPN Example—Customer Provides Internet Service

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:

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;
         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 VPN 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;
  }
}
```

Carrier-of-Carriers VPN Example—Customer Provides VPN Service

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, and a functioning MPLS path is required between these routers if they exchange VPN-IPv4 routes.

For configuration information see the following sections:

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 (see Figure 63 on page 604) 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-l {
      export attached;
      peer-as 21;
      neighbor 192.168.197.197;
    }
  }
}
policy-options {
  policy-statement attached {
    from protocol direct;
    then accept;
  }
}
```

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 the *Junos Feature Guide* on the product documentation page of the Juniper Networks website, located at **http://www.juniper.net/**.

CHAPTER 25

Summary of Interprovider and Carrier-of-Carriers VPNs Configuration Statements

The following sections explain the configuration statements that apply specifically to hierarchical and recursive BGP and MPLS virtual private networks (VPNs). The statements are organized alphabetically.

labeled-unicast

Syntax

```
labeled-unicast {
  per-group-label;
  resolve-vpn;
  traffic-statistics {
    file filename <world-readable | no-world-readable>;
    interval seconds;
  }
}
```

Hierarchy Level

[edit logical-systems *logical-system-name* protocols bgp family inet],
[edit logical-systems *logical-system-name* protocols bgp group *group-name* family inet],
[edit protocols bgp family inet],
[edit protocols bgp group *group-name* family inet]

Release Information

Statement introduced before Junos OS Release 7.4.

Description

Advertise labeled routes from the **inet.0** VPN, and place labeled routes into the **inet.0** VPN. When the **labeled-unicast** statement is used, the local router automatically performs a next hop to self on all routes advertised into EBGP from IBGP and from IBGP to EBGP.

Options

resolve-vpn—(Optional) Store labeled routes in the **inet.3** routing table to resolve routes for a provider edge (PE) router located in a different autonomous system (AS). For a PE router to install a route in the VPN routing and forwarding (VRF) table, the next hop must resolve to a route stored in the **inet.3** routing table. This option is also used to configure inter-AS VPLS with MAC operations.

The other statements are explained separately.

Required Privilege Level

routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation

- Configuring Carrier-of-Carriers VPNs for Customers That Provide Internet Service on page 573
- Configuring Carrier-of-Carriers VPNs for Customers That Provide VPN Service on page 579
- Configuring Inter-AS VPLS with MAC Processing at the ASBR on page 514
- Configuring Interprovider VPNs on page 569

per-group-label

Syntax per-group-label;

Hierarchy Level [edit logical-systems *logical-system-name* protocols bgp family inet labeled-unicast],
[edit logical-systems *logical-system-name* protocols bgp group *group-name* family inet labeled-unicast],
[edit protocols bgp family inet labeled-unicast],
[edit protocols bgp group *group-name* family inet labeled-unicast]

Release Information Statement introduced before Junos OS Release 7.4.

Description Account for traffic from each customer separately by advertising separate labels for the same prefix to the peer routers in the BGP groups.

Required Privilege Level routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation
- Configuring BGP to Gather Interprovider and Carrier-of-Carriers VPNs Statistics on page 586

traffic-statistics

Syntax
```
traffic-statistics {
  file filename <world-readable | no-world-readable>;
  interval seconds;
}
```

Hierarchy Level [edit logical-systems *logical-system-name* protocols bgp family inet labeled-unicast],
[edit logical-systems *logical-system-name* protocols bgp group *group-name* family inet labeled-unicast],
[edit protocols bgp family inet labeled-unicast],
[edit protocols bgp group *group-name* family inet labeled-unicast]

Release Information Statement introduced before Junos OS Release 7.4.

Description Enable the collection of traffic statistics for interprovider or carrier-of-carriers VPNs.

Options **file *filename***—Specify a filename for the BGP labeled–unicast traffic statistics file. If you do not specify a filename, statistics are still collected but can only be viewed by using the **show bgp group traffic statistics *group-name*** command.

interval *seconds*—Specify how often BGP labeled-unicast traffic statistics are collected.

Required Privilege Level routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation
- Configuring BGP to Gather Interprovider and Carrier-of-Carriers VPNs Statistics on page 586

PART 7

Layer 2 Circuits

CHAPTER 26

Layer 2 Circuit Overview

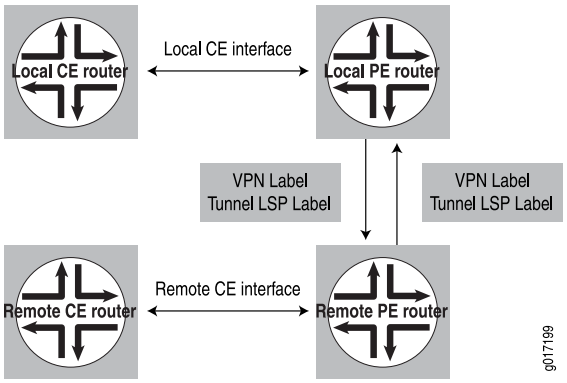
This chapter discusses the following topics:

Layer 2 Circuit Overview

A Layer 2 circuit is a point-to-point Layer 2 connection transported by means of MPLS or another tunneling technology on the service provider's network. A Layer 2 circuit is similar to a circuit cross-connect (CCC), except that multiple Layer 2 circuits can be transported over a single label-switched path (LSP) tunnel between two provider edge (PE) routers. In contrast, each CCC requires a dedicated LSP.

The Junos OS implementation of Layer 2 circuits supports only the remote form of a Layer 2 circuit; that is, a connection from a local customer edge (CE) router to a remote CE router. Figure 64 on page 633 illustrates the components of a Layer 2 circuit.

Figure 64: Components of a Layer 2 Circuit



The interfaces shown in Figure 64 on page 633 are logical interfaces. Packets are sent to the remote CE router by means of an egress virtual private network (VPN) label advertised by the remote PE router. The VPN label transits over either an RSVP or an LDP LSP (or

other type) tunnel to the remote PE router connected to the remote CE router. If you configure RSVP for Layer 2 circuits, you must also configure LDP.

Return traffic sent from the remote CE router to the local CE router uses an ingress VPN label advertised by the local PE router, which again transits over an RSVP and LDP LSP to the local PE router from the remote PE router. LDP is the signaling protocol used for advertising VPN labels.

Layer 2 Circuit Bandwidth Accounting and Call Admission Control

The sections that follow discuss Layer 2 circuit bandwidth accounting and call admission control (CAC):

Bandwidth Accounting and Call Admission Control Overview

Some network environments require that a certain level of service be guaranteed across the entire length of a path transiting a service provider's network. For Layer 2 circuits transiting an MPLS core network, a customer requirement might be to assure that guarantees for bandwidth and class of service (CoS) be maintained across the core network. For example, an Asynchronous Transfer Mode (ATM) circuit can provide service guarantees for each traffic class. A Layer 2 circuit configured to transport that ATM circuit across the network could be expected to provide the same service guarantees.

Providing this type of service guarantee requires the following:

- The LSPs in the MPLS core network must be able to provide service guarantees for bandwidth, rerouting, and route failures. You accomplish these guarantees by configuring multiclass LSPs. For more information about multiclass LSPs, see the *Junos OS MPLS Applications Configuration Guide*.
- The service guarantee must be maintained across the entire length of the link as it transits the service provider's network. Different Layer 2 circuits could have different bandwidth requirements. However, many Layer 2 circuits could be transported over the same E-LSP in the MPLS core network.
- CAC ensures that the LSP has sufficient bandwidth to accommodate the Layer 2 circuit. If there is not enough bandwidth over a particular LSP, the Layer 2 circuit is prevented from using that LSP.

Selecting an LSP Based on the Bandwidth Constraint

CAC of Layer 2 circuits is based on the bandwidth constraint. You must configure this constraint for each Layer 2 circuit interface. If there is a bandwidth constraint configured for a Layer 2 circuit, CAC bases the final selection of which LSP-forwarding next hop to use on the following:

- If multiple LSPs meet the bandwidth requirements, the first LSP found that can satisfy the bandwidth requirements for the Layer 2 circuit is selected.
- If there is more than one next hop mapped to the same LSP, then all the next hops that map to that LSP and pass CAC constraints are installed. This allows the Layer 2 circuit routes to restore themselves quickly in case of failure.
- The available bandwidth on the selected LSP is decremented by the bandwidth required for each Layer 2 circuit. Similarly, when the Layer 2 circuit route is changed or deleted (for example, when the route is disassociated from that particular LSP), the bandwidth on the corresponding LSP is incremented.
- There are no priorities among different Layer 2 circuits competing for the same LSP next hop in the core network.
- When an LSP's bandwidth changes, the Layer 2 circuits using that LSP repeat the CAC process again.

  If the LSP bandwidth increases, some Layer 2 circuits that were not established might now successfully resolve over the LSP. Similarly, if the bandwidth of the LSP decreases, some Layer 2 circuits that were previously up might now be declared down because of insufficient bandwidth on the LSP.
- When no LSP is found to meet the bandwidth requirements of the Layer 2 circuit, it is considered to be a CAC failure, and an error is reported.

LSP Path Protection and CAC

CAC can take into account LSPs that have been configured with an MPLS path protection feature, such as secondary paths, fast reroute, or node and link protection. CAC can consider the bandwidth available on these auxiliary links and can accept the backup connection as valid if the main connection fails. However, there are limitations on how the path protection feature must be configured to prevent CAC from taking down the Layer 2 circuit when the LSP it is using is switched to a backup route.

For more information about MPLS path protection features, see the *Junos OS MPLS Applications Configuration Guide*.

The sections that follow discuss the path protection features that can be used in conjunction with CAC and how they must be configured:

- Secondary Paths and CAC on page 635
- Fast Reroute and CAC on page 636
- Link and Node Protection and CAC on page 636

Secondary Paths and CAC

The following describes the ways in which secondary paths would interact with Layer 2 circuit CAC:

- If an LSP is configured with both primary and secondary paths, if the paths have the same bandwidth, and if this bandwidth is enough to accommodate the Layer 2 circuit, the Layer 2 circuit route installs both next hops in the forwarding table.

CAC allows the Layer 2 circuit to be switched to the secondary path if the primary path fails.

- If the LSP has primary and secondary paths configured with different bandwidths, each path must run through CAC independently. If the active path for that LSP passes CAC constraints successfully, then that next hop is installed and the corresponding LSP is selected to transport the Layer 2 circuit traffic. The LSP's secondary paths are then checked for CAC, and installed if there is sufficient bandwidth.

  However, if the active path for the LSP fails to meet the CAC constraints, then that LSP is not selected and the system looks for a different LSP to transport the Layer 2 circuit.

  For example, an LSP has an active primary path with 30 megabits of bandwidth and a secondary path with 10 megabits of bandwidth. The Layer 2 circuit requires 15 megabits of bandwidth. The secondary path fails CAC, and only the next hop corresponding to the primary path is installed for the Layer 2 circuit route. The path protection originally provided by the secondary path is no longer available.

Fast Reroute and CAC

No CAC is done for fast reroute detours. However, as long as the protected path satisfies the CAC bandwidth constraints, the detour next hop is also selected and installed.

Link and Node Protection and CAC

You can configure CAC on Layer 2 circuit-based LSPs with bandwidth constraints and also enable link and node protection. However, if the primary LSP fails, CAC might not be applied to the bypass LSP, meaning the bypass LSP might not meet the bandwidth constraint for the Layer 2 circuit. To minimize the risk of loosing traffic, the Layer 2 circuit continues to use the non-CAC bypass LSP while an attempt is made to establish a new Layer 2 circuit route over an LSP that does support CAC.

Layer 2 Circuits Trunk Mode

Using Layer 2 circuit trunk mode, you can configure Layer 2 circuits to carry ATM trunks, providing a way to link ATM switches over an MPLS core network.

Layer 2 circuit trunk mode allows you to configure the following CoS features:

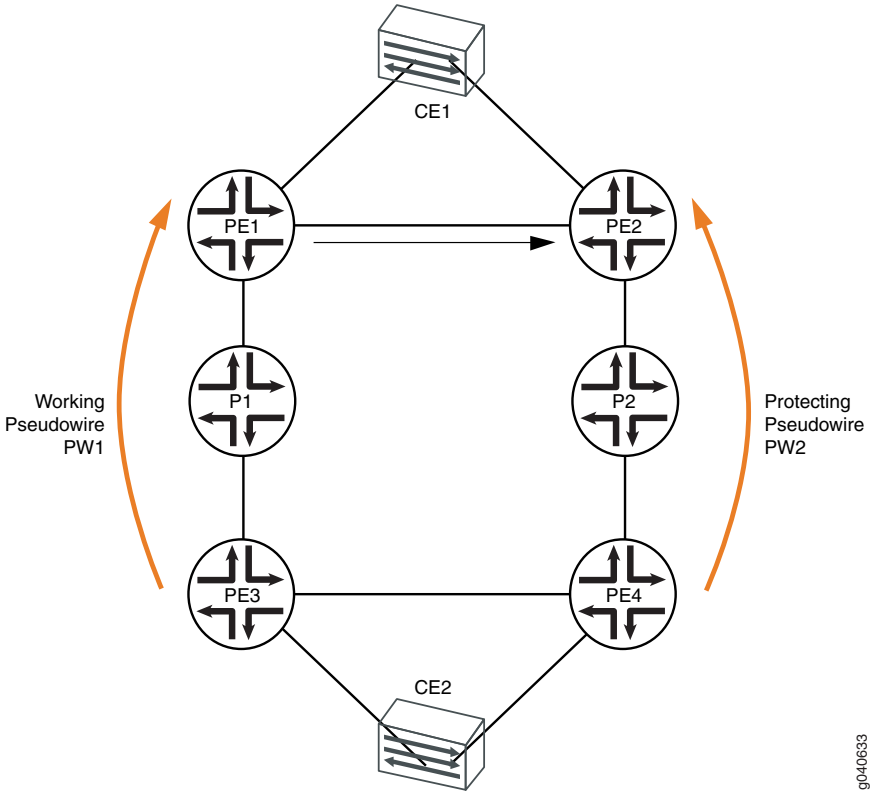
- CoS queues in Layer 2 circuit trunk mode—For ATM2 IQ interfaces, you can configure ATM CoS queues for Layer 2 circuit trunk mode.
- Layer 2 circuit trunk mode scheduling—For ATM2 IQ interfaces configured to use Layer 2 circuit trunk mode, you can share a scheduler among 32 trunks on an ATM port.
- Two early packet discard (EPD) thresholds per queue—For ATM2 IQ interfaces configured to use Layer 2 circuit trunk mode, you can set two EPD thresholds that depend on the packet-loss priorities (PLPs) of the packets.

For a detailed overview and configuration documentation, see the *Junos OS Network Interfaces Configuration Guide* and *Junos OS Class of Service Configuration Guide*.

Egress Protection LSPs for Layer 2 Circuits

An egress protection LSP provides link protection for link between PE routers and CE devices as illustrated in Figure 65 on page 637.

Figure 65: Egress protection LSP



Device CE1 is multihomed to router PE1 and router PE2. Device CE2 is multihomed to router PE3 and router PE4. There are two paths connecting devices CE1 and CE2. The working path is CE2-PE3-P1-PE1-CE1, using pseudowire PW1. The protecting path is CE2-PE3-P2-PE2-CE1, using pseudowire PW2. Normally, traffic flows through the working path. When the end-to-end OAM between devices CE1 and CE2 detects a failure on the working path, traffic will be switched from the working path to the protecting path.

In the topology shown in Figure 65 on page 637, if there was a link or node failure in the core network (for example, a link failure from router P1 to PE1, from router PE3 to P1, or a node failure of router P1), MPLS fast reroute can be triggered on the transport LSPs between router PE3 and router PE1 to repair the connection within tens of milliseconds. Egress protection LSPs address the problem of when a link failure occurs at the edge of the network (for example, a link failure on router PE1 to device CE1).

An egress protection LSP has been configured from router PE1 to router PE2. In the event of a link failure between router PE1 and device CE1, traffic can be switched to the egress

protection LSP. Traffic from device CE2 can now be routed through path PE3-P1-PE1-PE2 to reach device CE1.

Layer 2 Circuit Standards

The Junos OS substantially supports the following Layer 2 circuit standards:

- RFC 4447, *Pseudowire Setup and Maintenance Using the Label Distribution Protocol (LDP)* (except section 5.3)
- RFC 4448, *Encapsulation Methods for Transport of Ethernet over MPLS Networks*
- Internet draft draft-martini-l2circuit-encap-mpls-11.txt, *Encapsulation Methods for Transport of Layer 2 Frames Over IP and MPLS Networks* (expires August 2006)

The Junos OS has the following exceptions:

- A packet with a sequence number of 0 is treated as out of sequence.
  - Any packet that does not have the next incremental sequence number is considered out of sequence.
  - When out-of-sequence packets arrive, the expected sequence number for the neighbor is set to the sequence number in the Layer 2 circuit control word.
- Internet draft draft-martini-l2circuit-trans-mpls-19.txt, *Transport of Layer 2 Frames Over MPLS* (expires September 2006).

These drafts are available on the IETF website at **http://www.ietf.org/**.

CHAPTER 27

Configuring Layer 2 Circuits

This chapter describes how to configure Layer 2 circuits, discussing the following topics:

Introduction to Configuring Layer 2 Circuits

To configure a Layer 2 circuit, include the **l2circuit** statement:

```
l2circuit {
    local-switching {
        interface interface-name {
            description text;
            end-interface {
                interface interface-name;
                protect-interface interface-name;
            }
            ignore-mtu-mismatch;
            protect-interface interface-name;
        }
    }
    neighbor address {
        interface interface-name {
            community community-name;
            (control-word | no-control-word);
            description text;
            encapsulation-type type;
            mtu mtu-number;
            protect-interface interface-name;
```

```
        psn-tunnel-endpoint address;
        static {
          incoming-label label;
          outgoing-label label;
        }
        virtual-circuit-id identifier;
      }
    }
    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 protocols]**
- **[edit logical-systems** *logical-system-name* **protocols]**

Configuring Interfaces for Layer 2 Circuits

The following sections describe how to configure interfaces for Layer 2 circuits:

- Configuring the Address for the Neighbor of the Layer 2 Circuit on page 640
- Configuring the Neighbor Interface for the Layer 2 Circuit on page 641
- Configuring the Interface Encapsulation Type for Layer 2 Circuits on page 648
- Configuring ATM2 IQ Interfaces for Layer 2 Circuits on page 648

Configuring the Address for the Neighbor of the Layer 2 Circuit

All the Layer 2 circuits using a particular remote PE router designated for remote CE routers are listed under the **neighbor** statement (“neighbor” designates the PE router). Each neighbor is identified by its IP address and is usually the end-point destination for the label-switched path (LSP) tunnel transporting the Layer 2 circuit.

To configure a PE router as a neighbor for a Layer 2 circuit, specify the neighbor address using the **neighbor** statement:

```
neighbor address {
  ...
}
```

You can include this statement at the following hierarchy levels:

- **[edit protocols l2circuit]**
- **[edit logical-systems** *logical-system-name* **protocols l2circuit]**

Configuring the Neighbor Interface for the Layer 2 Circuit

Each Layer 2 circuit is represented by the logical interface connecting the local provider edge (PE) router to the local customer edge (CE) router. This interface is tied to the Layer 2 circuit neighbor configured in "Configuring the Address for the Neighbor of the Layer 2 Circuit" on page 640.

To configure the interface for a Layer 2 circuit neighbor, include the **interface** statement:

```
interface interface-name {
  bandwidth (bandwidth | ctnumber bandwidth);
  community community-name;
  (control-word | no-control-word);
  description text;
  encapsulation-type type;
  ignore-encapsulation-mismatch;
  ignore-mtu-mismatch;
  mtu mtu-number;
  protect-interface interface-name;
  pseudowire-status-tlv;
  psn-tunnel-endpoint address;
  virtual-circuit-id identifier;
}
```

You can include this statement at the following hierarchy levels:

- **[edit protocols l2circuit neighbor *address*]**
- **[edit logical-systems *logical-system-name* protocols l2circuit neighbor *address*]**

The following sections describe how to configure the interface for the Layer 2 circuit neighbor:

Configuring a Community for the Layer 2 Circuit

To configure a community for a Layer 2 circuit, include the **community** statement:

```
community community-name;
```

You can include this statement at the following hierarchy levels:

- **[edit protocols l2circuit neighbor *address* interface *interface-name*]**

- **[edit logical-systems *logical-system-name* protocols l2circuit neighbor *address* interface *interface-name*]**

For information about how to configure a routing policy for a Layer 2 circuit, see "Configuring Policies for Layer 2 Circuits" on page 651.

Configuring the Control Word for Layer 2 Circuits

To emulate the virtual circuit (VC) encapsulation for Layer 2 circuits, a 4-byte control word is added between the Layer 2 protocol data unit (PDU) being transported and the VC label that is used for demultiplexing. For most protocols, a null control word consisting of all zeroes is sent between Layer 2 circuit neighbors.

However, individual bits are available in a control word that can carry Layer 2 protocol control information. The control information is mapped into the control word, which allows the header of a Layer 2 protocol to be stripped from the frame. The remaining data and control word can be sent over the Layer 2 circuit, and the frame can be reassembled with the proper control information at the egress point of the circuit.

The following Layer 2 protocols map Layer 2 control information into special bit fields in the control word:

- Frame Relay—The control word supports the transport of discard eligible (DE), forward explicit congestion notification (FECN), and backward explicit congestion notification (BECN) information. For configuration information, see "Configuring the Control Word for Frame Relay Interfaces" on page 643.
- ATM AAL5 mode—The control word supports the transport of sequence number processing, ATM cell loss priority (CLP), and explicit forward congestion indication (EFCI) information. When you configure an AAL5 mode Layer 2 circuit, the control information is carried by default and no additional configuration is needed.
- ATM cell-relay mode—The control word supports sequence number processing only. When you configure a cell-relay mode Layer 2 circuit, the sequence number information is carried by default and no additional configuration is needed.

The Junos OS implementation of sequence number processing for ATM cell-relay mode and AAL5 mode is not the same as that described in Sec. 3.1.2 of the IETF draft *Encapsulation Methods for Transport of Layer 2 Frames Over IP and MPLS Networks*. The differences are as follows:

- A packet with a sequence number of 0 is considered as out of sequence.
- A packet that does not have the next incremental sequence number is considered out of sequence.
- When out-of-sequence packets arrive, the sequence number in the Layer 2 circuit control word increments by one and becomes the expected sequence number for the neighbor.

The following sections discuss how to configure the control word for Layer 2 circuits:

- Configuring the Control Word for Frame Relay Interfaces on page 643
- Disabling the Control Word for Layer 2 Circuits on page 643

Configuring the Control Word for Frame Relay Interfaces

On interfaces with Frame Relay CCC encapsulation, you can configure Frame Relay control bit translation to support Frame Relay services over IP and MPLS backbones by using CCC, Layer 2 VPNs, and Layer 2 circuits. When you configure translation of Frame Relay control bits, the bits are mapped into the Layer 2 circuit control word and preserved across the IP or MPLS backbone.

For information about how to configure the control bits, see the *Junos OS Network Interfaces Configuration Guide* and the *Junos OS Feature Guide*.

Disabling the Control Word for Layer 2 Circuits

The Junos OS can typically determine whether a neighboring router supports the control word. However, if you want to explicitly disable its use on a specific interface, include the **no-control-word** statement:

```
no-control-word;
```

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

Configuring the Encapsulation Type for the Layer 2 Circuit Neighbor Interface

The encapsulation type you configure for each Layer 2 circuit neighbor interface varies depending on which Layer 2 protocol you choose to configure. You do not need to use the same encapsulation at the ingress and egress of the Layer 2 circuit neighbor interface if you configure any of the following encapsulation types:

- **atm-aal5**—Asynchronous Transfer Mode (ATM) Adaptation Layer (AAL5)
- **atm-cell**—ATM cell relay
- **atm-cell-port-mode**—ATM cell relay port promiscuous mode
- **atm-cell-vc-mode**—ATM virtual circuit (VC) cell relay nonpromiscuous mode
- **atm-cell-vp-mode**—ATM virtual path (VP) cell relay promiscuous mode
- **cesop**—CESOP-based Layer 2 circuit
- **cisco-hdlc**—Cisco Systems-compatible High-Level Data Link Control (HDLC)
- **ethernet**—Ethernet
- **ethernet-vlan**—Ethernet virtual LAN (VLAN)
- **frame-relay**—Frame Relay
- **frame-relay-port-mode**—Frame Relay port mode
- **interworking**—Layer 2.5 interworking
- **ppp**—Point-to-Point Protocol (PPP)

- **satsop-e1**—SATSOP-E1-based Layer 2 circuit
- **satsop-e3**—SATSOP-E3-based Layer 2 circuit
- **satsop-t1**—SATSOP-T1-based Layer 2 circuit
- **satsop-t3**—SATSOP-T3-based Layer 2 circuit

If you configure different encapsulation types at the ingress and egress of the Layer 2 circuit neighbor interface, you need to configure a translational cross-connect (TCC) encapsulation type. For more information, see "Configuring TCC Encapsulation for Layer 2 VPNs and Layer 2 Circuits" on page 85.

Specify the encapsulation type by including the **encapsulation-type** statement:

encapsulation-type *type*;

You can include this statement at the following hierarchy levels:

- **[edit protocols l2circuit neighbor** *address* **interface** *interface-name***]**
- **[edit logical-systems** *logical-system-name* **protocols l2circuit neighbor** *address* **interface** *interface-name***]**

Enabling the Layer 2 Circuit When the Encapsulation Does Not Match

You can configure the Junos OS to allow a Layer 2 circuit to be established even though the encapsulation configured on the CE device interface does not match the encapsulation configured on the Layer 2 circuit interface by including the **ignore-encapsulation-mismatch** statement. You can configure the **ignore-encapsulation-mismatch** statement for the connection to the remote connection by including the statement at the **[edit protocols l2circuit neighbor** *address* **interface** *interface-name***]** hierarchy level or for the local connection by including this statement at the **[edit protocols l2circuit local-switching interface** *interface-name***]** hierarchy level.

ignore-encapsulation-mismatch;

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

Configuring the MTU for the Layer 2 Circuit Neighbor Interface

The following sections describe how to configure the MTU for the Layer 2 circuit neighbor interface:

Enabling the Layer 2 Circuit When the MTU Does Not Match

You can configure the Junos OS to allow a Layer 2 circuit to be established even though the MTU configured on the PE router does not match the MTU configured on the remote PE router by including the **ignore-mtu-mismatch** statement:

ignore-mtu-mismatch;

You can include this statement at the following hierarchy levels:

- **[edit protocols l2circuit neighbor** *address* **interface** *interface-name***]**
- **[edit logical-systems** *logical-system-name* **protocols l2circuit neighbor** *address* **interface** *interface-name***]**

Configuring the MTU Advertised for a Layer 2 Circuit

By default, the MTU used to advertise a Layer 2 circuit is determined by taking the interface MTU for the associated physical interface and subtracting the encapsulation overhead for sending IP packets based on the encapsulation.

However, encapsulations that support multiple logical interfaces (and multiple Layer 2 circuits) rely on the same interface MTU (since they are all associated with the same physical interface). This can prove to be a limitation for VLAN Layer 2 circuits using the same Ethernet interface or for Layer 2 circuit DLCIs using the same Frame Relay interface.

This can also affect multivendor environments. For example, if you have three PE devices supplied by different vendors and one of the devices only supports an MTU of 1500, even if the other devices support larger MTUs you must to configure the MTU as 1500 (the smallest MTU of the three PE devices).

You can explicitly configure which MTU is advertised for a Layer 2 circuit, even if the Layer 2 circuit is sharing a physical interface with other Layer 2 circuits. When you explicitly configure an MTU for a Layer 2 circuit, be aware of the following:

- An explicitly configured MTU is signaled to the remote PE device. The configured MTU is also compared to the MTU received from the remote PE device. If there is a conflict, the Layer 2 circuit is taken down.
- If you configure an MTU for an ATM cell relay interface on an ATM II PIC, the configured MTU is used to compute the cell bundle size advertised for that Layer 2 circuit, instead of the default interface MTU.
- A configured MTU is used only in the control plane. It is not enforced in the data plane. You need to ensure that the CE device for a given Layer 2 circuit uses the correct MTU for data transmission.

To configure the MTU for a Layer 2 circuit, include the **mtu** statement:

mtu *mtu-number*;

You can include this statement at the following hierarchy levels:

- **[edit protocols l2circuit neighbor** *address* **interface** *interface-name***]**
- **[edit logical-systems** *logical-system-name* **protocols l2circuit neighbor** *address* **interface** *interface-name***]**

Configuring the Protect Interface

You can configure a protect interface for the logical interface linking a virtual circuit to its destination, whether the destination is remote or local. A protect interface provides a backup for the protected interface in case of failure. Network traffic uses the primary

interface only so long as the primary interface functions. If the primary interface fails, traffic is switched to the protect interface. The protect interface is optional.

To configure the protect interface, include the **protect-interface** statement:

protect-interface *interface-name*;

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

For an example of how to configure a protect interface for a Layer 2 circuit, see "Introduction to Layer 2 Circuit Protect Interfaces Example" on page 659.

Configuring the Pseudowire Status TLV

The pseudowire status type length variable (TLV) is used to communicate the status of a pseudowire back and forth between two PE routers. For Layer 2 circuit configurations, you can configure the PE router to negotiate the pseudowire with its neighbor using the pseudowire status TLV. This same functionality is also available for LDP VPLS neighbor configurations. The pseudowire status TLV is configurable for each pseudowire connection and is disabled by default. The pseudowire status negotiation process assures that a PE router reverts back to the label withdraw method for pseudowire status if its remote PE router neighbor does not support the pseudowire status TLV.

Unlike the control word, a PE router's ability to support the pseudowire status TLV is communicated when the initial label mapping message is sent to its remote PE router. Once the PE router transmits its support for the pseudowire status TLV to its remote PE router, it includes the pseudowire status TLV in every label mapping message sent to the remote PE router. If you disable support for the pseudowire status TLV on the PE router, a label withdraw message is sent to the remote PE router and then a new label mapping message without the pseudowire status TLV follows.

To configure the pseudowire status TLV for the pseudowire to the neighbor PE router, include the **pseudowire-status-tlv** statement:

pseudowire-status-tlv;

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

Configuring Layer 2 Circuits over Both RSVP and LDP LSPs

You can configure two Layer 2 circuits between the same two routers, and have one Layer 2 circuit traverse an RSVP LSP and the other traverse an LDP LSP. To accomplish this, you need to configure two loopback addresses on the local router. You configure one of the loopback address for the Layer 2 circuit traversing the RSVP LSP. You configure the other loopback address to handle the Layer 2 circuit traversing the LDP LSP. For information about how to configure multiple loop back interfaces, see "Configuring Logical Units on the Loopback Interface for Routing Instances in Layer 3 VPNs" on page 193.

You also need to configure a packet switched network (PSN) tunnel endpoint for one of the Layer 2 circuits. It can be either the Layer 2 circuit traversing the RSVP LSP or the one traversing the LDP LSP. The PSN tunnel endpoint address is the destination address for the LSP on the remote router.

To configure the address for the PSN tunnel endpoint, include the **psn-tunnel-endpoint** statement:

psn-tunnel-endpoint *address*;

You can include this statement at the following hierarchy levels:

- **[edit logical-systems** *logical-system-name* **protocols l2circuit neighbor** *address* **interface** *interface-name***]**
- **[edit protocols l2circuit neighbor** *address* **interface** *interface-name***]**

By default, the PSN tunnel endpoint for a Layer 2 circuit is identical to the neighbor address, which is also the same as the LDP neighbor address.

The tunnel endpoints on the remote router do not need to be loopback addresses.

Example: PSN Tunnel Endpoint

The following example illustrates how you might configure a PSN tunnel endpoint:

```
[edit protocols l2circuit]
neighbor 10.255.0.6 {
   interface t1-0/2/2.0 {
      psn-tunnel-endpoint 20.20.20.20;
      virtual-circuit-id 1;
   }
   interface t1-0/2/1.0 {
      virtual-circuit-id 10;
   }
}
```

The Layer 2 circuit configured for the **t1-0/2/2.0** interface resolves in the inet3 routing table to **20.20.20.20**. This could be either an RSVP route or a static route with an LSP next hop.

Configuring the Virtual Circuit ID

You configure a virtual circuit ID on each interface. Each virtual circuit ID uniquely identifies the Layer 2 circuit among all the Layer 2 circuits to a specific neighbor. The key to identifying a particular Layer 2 circuit on a PE router is the neighbor address and the virtual circuit ID. An LDP-FEC-to-label binding is associated with a Layer 2 circuit based on the virtual circuit ID in the FEC and the neighbor that sent this binding. The LDP-FEC-to-label binding enables the dissemination of the VPN label used for sending traffic on that Layer 2 circuit to the remote CE device.

You also configure a virtual circuit ID for each redundant pseudowire. A redundant pseudowire is identified by the backup neighbor address and the virtual circuit ID. For more information, see “Configuring Pseudowire Redundancy on the PE Router” on page 34.

To configure the virtual circuit ID, include the **virtual-circuit-id** statement:

virtual-circuit-id *identifier*;

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

Configuring the Interface Encapsulation Type for Layer 2 Circuits

The Layer 2 encapsulation type is carried in the LDP forwarding equivalence class (FEC). You can configure either circuit cross-connect (CCC) or translational cross-connect (TCC) encapsulation types for Layer 2 circuits. For more information, see the *Junos OS MPLS Applications Configuration Guide*.

To configure the interface encapsulation for a Layer 2 circuit, include the **encapsulation-type** statement:

encapsulation-type *encapsulation-type*;

You can include this statement at the following hierarchy levels:

- **[edit interfaces** *interface-name***]**
- **[edit logical-systems** *logical-system-name* **interfaces** *interface-name***]**

Configuring ATM2 IQ Interfaces for Layer 2 Circuits

You can configure Asynchronous Transfer Mode 2 (ATM2) intelligent queuing (IQ) interfaces for Layer 2 circuits by using Layer 2 circuit ATM Adaptation Layer 5 (AAL5) transport mode, Layer 2 circuit ATM cell relay mode, and the Layer 2 circuit ATM trunk mode.

The configuration statements are as follows:

- **atm-l2circuit-mode aal5**
- **atm-l2circuit-mode cell**
- **atm-l2circuit-mode trunk**

For more information about these statements, see the *Junos OS System Basics Configuration Guide*. For more information about how to configure ATM2 IQ interfaces, see the *Junos OS Network Interfaces Configuration Guide*.

The Junos OS implementation of sequence number processing for Layer 2 circuit ATM cell relay mode and Layer 2 circuit AAL5 mode differs from that described in the Internet draft draft-martini-l2circuit-encap-mpls-11.txt, *Encapsulation Methods for Transport of Layer 2 Frames over MPLS Networks* (expires August 2006).

The Junos OS implementation has the following differences:

1. A packet with a sequence number of 0 is treated as out of sequence.
2. A packet that does not have the next incremental sequence number is considered out of sequence.

When out-of-sequence packets arrive, the expected sequence number for the neighbor is set to the sequence number in the Layer 2 circuit control word.

Configuring Local Interface Switching in Layer 2 Circuits

You can configure a virtual circuit entirely on the local router, terminating the circuit on a local interface. Possible uses for this feature include being able to enable switching between Frame Relay DLCIs.

To configure a virtual circuit to terminate locally, include the **local-switching** statement:

```
local-switching {
  interface interface-name {
    description text;
    end-interface {
      interface interface-name;
      protect-interface interface-name;
    }
    ignore-mtu-mismatch;
    protect-interface interface-name;
  }
}
```

You can include this statement at the following hierarchy levels:

- **[edit protocols l2circuit]**
- **[edit logical-systems *logical-system-name* protocols l2circuit]**

The following sections describe how to configure local interface switching:

- Configuring the Interfaces for the Local Interface Switch on page 649
- Enabling Local Interface Switching When the MTU Does Not Match on page 650

Configuring the Interfaces for the Local Interface Switch

Local interface switching requires you to configure at least two interfaces:

- Starting interface—Include the **interface** statement at the **[edit protocols l2circuit local-switching]** hierarchy level.
- Ending interface—Include the **end-interface** statement at the **[edit protocols l2circuit local-switching interface *interface-name*]** hierarchy level.

You can also configure virtual circuit interface protection for each local interface:

- Protect interface for the starting interface—Include the **protect-interface** statement at the **[edit protocols l2circuit local-switching interface *interface-name*]** hierarchy level.
- Protect interface for the ending interface—Include the **protect-interface** statement at the **[edit protocols l2circuit local-switching interface *interface-name* end-interface]** hierarchy level.

For more information about how to configure protect interfaces, see “Configuring the Protect Interface” on page 645.

Enabling Local Interface Switching When the MTU Does Not Match

You can configure a local switching interface to ignore the MTU configuration set for the associated physical interface. This enables you to bring up a circuit between two logical interfaces that are defined on physical interfaces with different MTU values.

To configure the local switching interface to ignore the MTU configured for the physical interface, include the **ignore-mtu-mismatch** statement:

```
ignore-mtu-mismatch;
```

You can include this statement at the following hierarchy levels:

- **[edit protocols l2circuit local-switching interface *interface-name*]**
- **[edit logical-systems *logical-system-name* protocols l2circuit local-switching interface *interface-name*]**

Configuring LDP for Layer 2 Circuits

Use LDP as the signaling protocol to advertise ingress labels to the remote PE routers. When configured, LDP examines the Layer 2 circuit configuration and initiates extended neighbor discovery for all the Layer 2 circuit neighbors (for example, remote PEs). This process is similar to how LDP works when tunneled over RSVP. You must run LDP on the **lo0.0** interface for extended neighbor discovery to function correctly.

For detailed information about how to configure LDP, see the *Junos OS MPLS Applications Configuration Guide*.

Configuring Static Layer 2 Circuits

You can configure static Layer 2 circuit pseudowires. Static pseudowires are designed for networks that do not support LDP or do not have LDP enabled. You configure a static pseudowire by configuring static values for the in and out labels needed to enable a pseudowire connection. The **ignore-mtu-mismatch**, **ignore-vlan-id**, and **ignore-encapsulation-mismatch** statements are not relevant for static pseudowire configurations since the peer router cannot forward this information.

When you configure static pseudowires, you need to manually compare the encapsulation, TDM bit rate, and control word of the router with the remote peer router and ensure that they match, otherwise the static pseudowire might not work.

To configure static Layer 2 circuit pseudowires, include the **static** statement:

```
static {
   incoming-label label;
   outgoing-label label;
}
```

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

You can configure a static pseudowire as a standalone Layer 2 circuit or in conjunction with a redundant pseudowire. You configure the static pseudowire statement at the **[edit protocols l2circuit neighbor *address* interface *interface-name*]** hierarchy level. You configure the redundant pseudowire at the **[edit protocols l2circuit neighbor *address* interface *interface-name* backup-neighbor *neighbor*]** hierarchy level. If you configure a static pseudowire to a neighbor and also configure a redundant pseudowire, the redundant pseudowire must also be static.

For information about how to configure redundant pseudowires, see "Configuring Redundant Pseudowires for Layer 2 Circuits and VPLS" on page 34.

Configuring Policies for Layer 2 Circuits

You can configure Junos routing policies to control the flow of packets over Layer 2 circuits. This capability allows you to provide different level of service over a set of equal-cost Layer 2 circuits. For example, you can configure a circuit for high-priority traffic, a circuit for average-priority traffic, and a circuit for low-priority traffic. By configuring Layer 2 circuit policies, you can ensure that higher-value traffic has a greater likelihood of reaching its destination.

The following sections explain how to configure Layer 2 circuit policies:

Configuring the Layer 2 Circuit Community

To configure a community for Layer 2 circuits, include the **community** statement.

```
community community-name {
    members [ community-ids ];
}
```

You can include this statement at the following hierarchy levels:

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

name identifies the community or communities.

community-ids identifies the type of community or extended community:

- A normal community uses the following community ID format:

  as-number*:*community-value

  as-number is the autonomous system (AS) number of the community member.

  community-value is the identifier of the community member. It can be a number from 0 through 65,535.

- An extended community uses the following community ID format:

type:*administrator*:*assigned-number*

type is the type of target community. The target community identifies the route's destination.

administrator is either an AS number or an IP version 4 (IPv4) address prefix, depending on the type of community.

assigned-number identifies the local provider.

You also need to configure the community for the Layer 2 circuit interface; see "Configuring a Community for the Layer 2 Circuit" on page 641.

Configuring the Policy Statement for the Layer 2 Circuit Community

To configure a policy to send community traffic over a specific LSP, include the **policy-statement** statement:

```
policy-statement policy-name {
  term term-name {
    from community community-name;
    then {
      install-nexthop (except | lsp lsp-name | lsp-regex lsp-regular-expression);
      accept;
    }
  }
}
```

You can include this statement at the following hierarchy levels:

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

To prevent the installation of any matching next hops, include the **install-nexthop** statement with the **except** option:

```
install-nexthop except;
```

You can include this statement at the following hierarchy levels:

- **[edit policy-options policy-statement** *policy-name* **term** *term-name* **then]**
- **[edit logical-systems** *logical-system-name* **policy-options policy-statement** *policy-name* **term** *term-name* **then]**

To assign traffic from a community to a specific LSP, include the **install-nexthop** statement with the **lsp** *lsp-name* option and the **accept** statement:

```
install-nexthop lsp lsp-name;
accept;
```

You can include these statements at the following hierarchy levels:

- **[edit policy-options policy-statement** *policy-name* **term** *term-name* **then]**
- **[edit logical-systems** *logical-system-name* **policy-options policy-statement** *policy-name* **term** *term-name* **then]**

You can also use a regular expression to select an LSP from a set of similarly named LSPs for the **install-nexthop** statement. To configure a regular expression, include the **install-nexthop** statement with the **lsp-regex** option and the **accept** statement:

```
install-nexthop lsp-regex lsp-regular-expression;
accept;
```

You can include these statements at the following hierarchy levels:

- **[edit policy-options policy-statement** *policy-name* **term** *term-name* **then]**
- **[edit logical-systems** *logical-system-name* **policy-options policy-statement** *policy-name* **term** *term-name* **then]**

Example: Configuring a Policy for a Layer 2 Circuit Community

The following example illustrates how you might configure a regular expression in a Layer 2 circuit policy. You create three LSPs to handle gold-tier traffic from a Layer 2 circuit. The LSPs are named **alpha-gold**, **beta-gold**, and **delta-gold**. You then include the **install-nexthop** statement with the **lsp-regex** option with the LSP regular expression **.*-gold** at the **[edit policy-options policy-statement** *policy-name* **term** *term-name* **then]** hierarchy level:

```
[edit policy-options]
policy-statement gold-traffic {
   term to-gold-LSPs {
      from community gold;
      then {
         install-nexthop lsp-regex .*-gold;
         accept;
      }
   }
}
```

The community **gold** Layer 2 circuits can now use any of the **-gold** LSPs. Given equal utilization across the three **-gold** LSPs, LSP selection is made at random.

You need to apply the policy to the forwarding table. To apply a policy to the forwarding table, configure the **export** statement at the **[edit routing-options forwarding-table]** hierarchy level:

```
[edit routing-options forwarding-table]
export policy-name;
```

Verifying the Layer 2 Circuit Policy Configuration

To verify that you have configured a policy for the Layer 2 circuit, issue the **show route table mpls detail** command. It should display the community for ingress routes that corresponds to the Layer 2 circuits, as shown by the following example:

```
user@host> show route table mpls detail
so-1/0/1.0 (1 entry, 1 announced)
*L2VPN  Preference: 7
Next hop: via so-1/0/0.0 weight 1, selected
Label-switched-path to-community-gold
Label operation: Push 100000 Offset: -4
Next hop: via so-1/0/0.0 weight 1
```

```
Label-switched-path to-community-silver
Label operation: Push 100000 Offset: -4
Protocol next hop: 10.255.245.45
Push 100000 Offset: -4
Indirect next hop: 85333f0 314
State: <Active Int>
Local AS:   100
Age: 22
Task: Common L2 VC
Announcement bits (2): 0-KRT 1-Common L2 VC
AS path: I
Communities: 100:1
```

For more information about how to configure routing policies, see the *Junos OS Policy Framework Configuration Guide*.

Configuring ATM Trunking on Layer 2 Circuits

You can configure Layer 2 circuits to transport ATM traffic from directly connected ATM switches across an MPLS core network. Traffic from an ATM switch is received on the local PE router. The ATM cells are given an MPLS label and then sent across the MPLS network to the remote PE router. The receiving router removes the MPLS label from the ATM cell and then forwards the cell the receiving ATM switch.

NOTE: ATM trunking on Layer 2 circuits is supported only on T Series and M320 routers and ATM2 IQ PICs.

Figure 66: ATM Trunking on Layer 2 Circuits

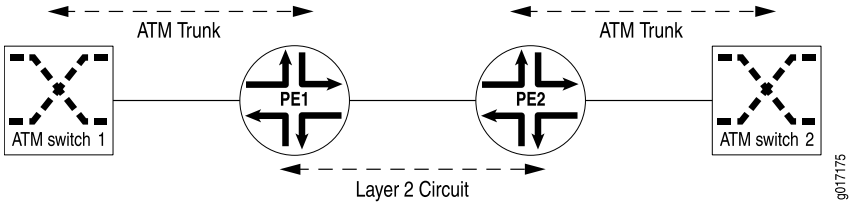


Figure 66 on page 654 illustrates how ATM switches could be linked together by a Layer 2 circuit. The PE1 Router is configured to receive ATM trunk traffic from ATM Switch 1. As each ATM cell is received on the PE1 Router, it is classified by means of the class-of-service (CoS) information in the cell header and then encapsulated as a labeled packet. The CoS information and cell loss priority (CLP) of the ATM cell are copied into the experimental (EXP) bits of the MPLS label. The labeled packet is then transported across the service provider network to the PE2 Router by means of a Layer 2 circuit.

On the PE2 Router, the label is removed and the plain ATM cell is forwarded to ATM Switch 2. The CoS and CLP are extracted from the EXP bits and are then used to select the correct output queue and determine whether the ATM cell should be dropped.

The ATM physical port on the router can support 32 logical trunks when network-to-network interface (NNI) is used and 8 logical trunks when user-to-network interface (UNI) is used. A trunk can carry traffic on 32 virtual path identifiers (VPIs), numbered 0 through 31. Each ATM trunk is associated with an MPLS label and a logical

interface. On the ingress router, one or more of these trunks are mapped to a Layer 2 circuit.

The configuration for the Layer 2 circuit between PE routers is conventional. Follow the procedures outlined in this chapter for configuring the circuit. However, there is some specific configuration you need to complete for the Layer 2 circuit to carry traffic from an ATM trunk.

First, enable ATM trunking for Layer 2 circuits. To enable ATM trunking for Layer 2 circuits, specify the **trunk** option for the **atm-l2circuit-mode** statement at the **[edit chassis fpc** *number* **pic** *number***]** hierarchy level:

```
[edit chassis fpc number pic number]
atm-l2circuit-mode trunk (uni | nni);
```

Specify the **uni** option for UNI trunks and the **nni** option for NNI trunks. The default option is **uni**.

You also need to configure each ATM trunk for a specific logical interface. Each ATM trunk has a trunk identifier in the range from 0 to 31. This configuration step is in addition to the typical configuration steps you follow related to configuring interfaces for Layer 2 circuits, as described in “Configuring Interfaces for Layer 2 Circuits” on page 640.

To associate a specific trunk identifier with a logical interface, include the **trunk-id** statement:

```
trunk-id number;
```

You can include this statement at the following hierarchy levels:

- **[edit interfaces** *interface-name* **unit** *number***]**
- **[edit logical-systems** *logical-system-name* **interfaces** *interface-name* **unit** *number***]**

Since ATM trunking is supported on ATM2 IQ PICs only, the only value you can configure for the **pic-type** statement is **atm2**. If you do not configure the **pic-type** statement but you do configure the **trunk** option for the **atm-l2circuit-mode** statement (at the **[chassis fpc** *number* **pic** *number***]** hierarchy level), the **pic-type** statement defaults to **atm2**.

Configuring Bandwidth Allocation and Call Admission Control in Layer 2 Circuits

You can configure bandwidth allocation and call admission control (CAC) on Layer 2 circuits. This feature is available for RSVP-signaled LSPs traversing an MPLS network.

When you enable bandwidth allocation on a Layer 2 circuit, attempts to establish an RSVP-signaled LSP are preceded by a check of the available bandwidth on the network. This check is the CAC. The available bandwidth is compared to the bandwidth requested by the LSP. If there is insufficient bandwidth, the Layer 2 circuit is not established and an error message is generated. To apply CAC to a Layer 2 circuit, a bandwidth constraint must be configured.

You can specify the bandwidth for a Layer 2 circuit without configuring a bandwidth for each class type (queue). To specify the bandwidth allocation for a Layer 2 circuit, include the **bandwidth** statement:

bandwidth *bandwidth*;

Specify the bandwidth in bits per second.

You can include this statement at the following hierarchy levels:

- **[edit protocols l2circuit neighbor** *address* **interface** *interface-name***]**
- **[edit logical-systems** *logical-system-name* **protocols l2circuit neighbor** *address* **interface** *interface-name***]**

Alternatively, you can configure the bandwidth for each class type on a Layer 2 circuit. If you use this type of configuration, you cannot simultaneously configure the nonclass type of bandwidth configuration for the Layer 2 circuit (the commit operation fails).

To configure the bandwidth for each class type on an Layer 2 circuit, include the **bandwidth** statement:

```
bandwidth {
    ct0 bandwidth;
    ct1 bandwidth;
    ct2 bandwidth;
    ct3 bandwidth;
}
```

You can include this statement at the following hierarchy levels:

- **[edit protocols l2circuit neighbor** *address* **interface** *interface-name***]**
- **[edit logical-systems** *logical-system-name* **protocols l2circuit neighbor** *address* **interface** *interface-name***]**

Specify the bandwidth for each class type in bits per second. It is not necessary to specify a bandwidth for all four class types.

Tracing Layer 2 Circuit Operations

To trace the creation of and changes to Layer 2 circuits, include the **traceoptions** statement:

```
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 protocols l2circuit]**
- **[edit logical-systems** *logical-system-name* **protocols l2circuit]**

Specify the following flags to trace the indicated operations on Layer 2 circuits:

- **connections**—Layer 2 circuit connections (events and state changes)
- **error**—Error conditions
- **FEC**—Layer 2 circuit advertisements received or sent using LDP
- **topology**—Layer 2 circuit topology changes caused by reconfiguration or advertisements received from other PE routers

CHAPTER 28

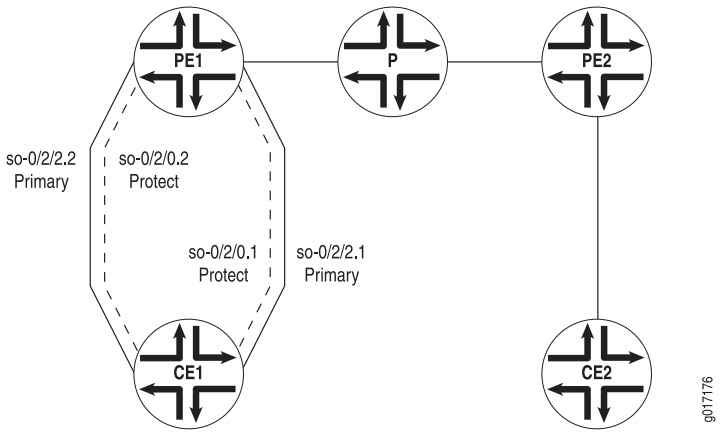
Layer 2 Circuits Examples

Introduction to Layer 2 Circuit Protect Interfaces Example

This example illustrates how you might configure a Layer 2 circuit with protect interfaces. Protect interfaces act as backups for their associated interfaces. The primary interface has priority over the protect interface and carries network traffic as long as it is functional. If the primary interface fails, the protect interface is activated. These interfaces can also share the same virtual path identifier (VPI) or virtual circuit identifier (VCI).

Figure 67 on page 659 shows the network topology used in this example.

Figure 67: Layer 2 Circuits Using Protect Interfaces



The following sections describe how to configure a Layer 2 circuit to use a protect interface:

- Configuring Router PE1 on page 659
- Configuring Router PE2 on page 661
- Configuring Router CE1 on page 663
- Configuring Router CE2 on page 663

Configuring Router PE1

Configure an interface for traffic to Router CE1 from Router PE1 at the **[edit interfaces]** hierarchy level:

```
[edit interfaces]
so-0/2/2 {
   description "Router CE1 so-0/2/2";
   no-keepalives;
   encapsulation frame-relay-ccc;
   unit 1 {
      encapsulation frame-relay-ccc;
      point-to-point;
      dlci 600;
   }
   unit 2 {
      encapsulation frame-relay-ccc;
      point-to-point;
      dlci 602;
   }
}
```

Configure an interface for traffic to Router CE1 from Router PE1 at the **[edit interfaces]** hierarchy level. Logical interface **so-0/2/0.2** acts as the protect interface for **so-0/2/2.2**, and logical interface **so-0/2/0.1** acts as the protect interface for **so-0/2/2.1**:

```
[edit interfaces]
so-0/2/0 {
   description "to Router CE1 so-0/3/0";
   no-keepalives;
   encapsulation frame-relay-ccc;
   unit 1 {
      encapsulation frame-relay-ccc;
      dlci 600;
   }
   unit 2 {
      encapsulation frame-relay-ccc;
      dlci 602;
   }
}
```

Configure an interface for traffic to Router PE2 from Router PE1 at the **[edit interfaces]** hierarchy level:

```
[edit interfaces]
so-0/2/1 {
   description "to Router PE2 so-1/0/1";
   unit 0 {
      family inet {
         address 100.100.40.22/32 {
            destination 100.100.40.23;
         }
      }
      family iso;
      family mpls;
   }
}
```

Configure an interface for traffic to Router PE2 from Router PE1 at the **[edit interfaces]** hierarchy level:

```
[edit interfaces]
so-0/2/3 {
   description "Router PE2 so-1/0/3";
   unit 0 {
      family inet;
      family iso;
      family mpls;
   }
   lo0 {
      unit 0 {
         family inet {
            address 127.0.0.1/32;
            address 10.100.40.200/32;
         }
         family iso {
            address 47.0005.80ff.f800.0000.0108.0001.1921.6800.4213.00;
         }
      }
   }
}
```

Configure the Layer 2 circuit by including the **l2circuit** statement at the **[edit protocols]** hierarchy level. The logical interfaces for the Layer 2 circuits and their corresponding protect interfaces are included here:

```
[edit protocols]
l2circuit {
   neighbor 10.100.40.210 {
      interface so-0/2/2.2 {
         protect-interface so-0/2/0.2;
         virtual-circuit-id 2;
         no-control-word;
      }
      interface so-0/2/2.1 {
         protect-interface so-0/2/0.1;
         virtual-circuit-id 1;
         no-control-word;
      }
   }
}
```

Configuring Router PE2

Configure an interface for traffic to Router CE2 from Router PE2:

```
[edit interfaces]
so-1/0/0 {
   description "to Router CE2 so-0/2/0";
   no-keepalives;
   encapsulation frame-relay-ccc;
   unit 1 {
      encapsulation frame-relay-ccc;
      point-to-point;
      dlci 700;
   }
   unit 2 {
```

```
    encapsulation frame-relay-ccc;
    point-to-point;
    dlci 702;
  }
}
```

Configure an interface for traffic to Router PE1 from Router PE2:

```
[edit interfaces]
so-1/0/1 {
  description "to Router PE1 so-0/2/1";
  unit 0 {
    family inet {
      address 100.100.40.23/32 {
        destination 100.100.40.22;
      }
    }
    family iso;
    family mpls;
  }
}
```

Configure an interface for traffic to Router PE1 from Router PE2:

```
[edit interfaces]
so-1/0/3 {
  description "to Router PE1 so-0/2/3";
  unit 0 {
    family inet;
    family iso;
    family mpls;
  }
}
lo0 {
  unit 0 {
    family inet {
      address 127.0.0.1/32;
      address 10.100.40.210/32;
    }
    family iso {
      address 47.0005.80ff.f800.0000.0108.0001.1921.6800.4216.00;
    }
  }
}
```

Configure the Layer 2 circuit at the **[edit protocols]** hierarchy level:

```
[edit protocols]
l2circuit {
  neighbor 10.100.40.200 {
    interface so-1/0/0.1 {
      virtual-circuit-id 1;
      no-control-word;
    }
    interface so-1/0/0.2 {
      virtual-circuit-id 2;
      no-control-word;
```

```
    }
  }
}
```

Configuring Router CE1

Configure an interface for traffic to Router PE1 from Router CE1:

```
[edit interfaces]
so-0/3/0 {
  description "to Router PE1 so-0/2/0";
  no-keepalives;
  encapsulation frame-relay;
  unit 1 {
    dlci 601;
    family inet {
      address 12.12.12.1/24;
    }
  }
}
```

Configure an interface for traffic to Router PE1 from Router CE1:

```
[edit interfaces]
so-0/3/1 {
  description "Router PE1 so-0/2/2";
  no-keepalives;
  encapsulation frame-relay;
  unit 0 {
    dlci 600;
    family inet {
      address 10.10.10.1/24;
      address 11.1.1.1/24;
    }
    family iso;
    family mpls;
  }
  unit 2 {
    dlci 602;
    family inet {
      address 13.13.13.1/24;
    }
  }
}
```

Configuring Router CE2

Configure an interface for traffic to Router PE2 from Router CE2:

```
[edit interfaces]
so-0/2/0 {
  description "to Router PE2 so-1/0/0";
  no-keepalives;
  encapsulation frame-relay;
  unit 1 {
    dlci 700;
    family inet {
```

```
        address 10.10.10.2/24;
        address 11.1.1.2/24;
        address 12.12.12.2/24;
      }
    }
    unit 2 {
      dlci 702;
      family inet {
        address 13.13.13.2/24;
      }
    }
  }
```

Example: Configuring an Egress Protection LSP for a Layer 2 Circuit

This example shows how to configure an egress protection LSP.

Requirements

Egress protection LSPs are supported on Juniper Networks MX Series routers only. This requirement applies to the PE routers facilitating the egress protection LSP.

Egress Protection LSP Overview

If there is a link or node failure in the core network, a protection mechanism such as MPLS fast reroute can be triggered on the transport LSPs between the PE routers to repair the connection within tens of milliseconds. An egress protection LSP addresses the problem of when a link failure occurs at the edge of the network (for example, a link failure between a PE router and a CE device). Egress protection LSPs do not address the problem of a node failure at the edge of the network (for example, a failure of a PE router). An egress protection LSP is an RSVP-signaled ultimate hop popping LSP.

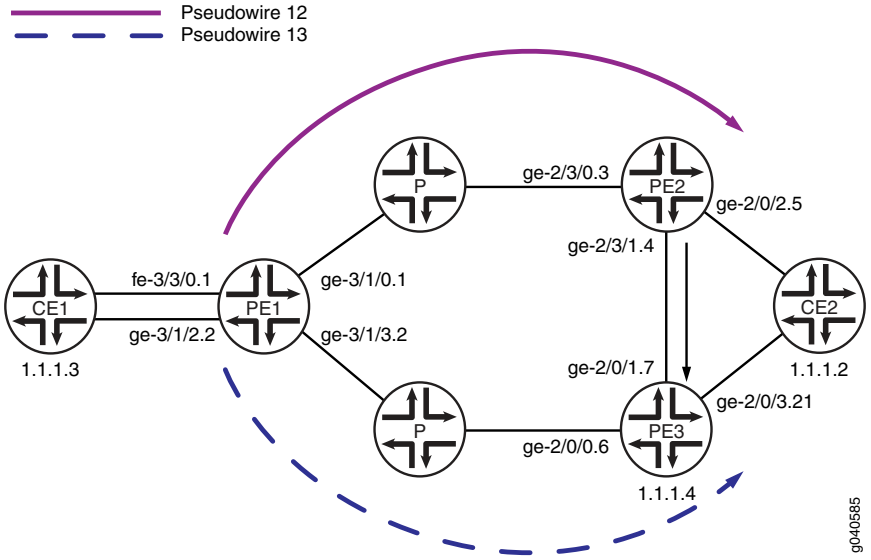
This example includes the following configuration concepts and statements that are unique to the configuration of an egress protection LSP:

- **context-identifier**—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.
- **egress-protection**—Configures the protector information for the protected Layer 2 circuit and configures the protector Layer 2 circuit at the **[edit protocols l2circuit]** hierarchy level. Configures an LSP as an egress protection LSP at the **[edit protocols mpls label-switched-path *lsp-name*]** hierarchy level. It also configures the context identifier at the **[edit protocols mpls]** hierarchy level.
- **protected-l2circuit**—Specifies which Layer 2 circuit is to be protected by the egress protect LSP. This statement includes the following sub-statements: **ingress-pe**,

egress-pe, and **virtual-circuit-id**. These sub-statements specify the address of the PE router at the ingress of the Layer 2 circuit, the address of the PE router at the egress of the Layer 2 circuit, and the Layer 2 circuit's identifier respectively.

- **protector-interface**—Specify the interface used by the egress protection LSP. In the event of a local link failure to a CE device, the egress protect LSP uses the interface specified to communicate with the protector PE router.
- **protector-pe**—Specify the IPv4 address of the protector PE router. The protector PE router must have a connection to the same CE device as the protected PE router for the egress protect LSP to function. This statement includes the following sub-statements: **context-identifier** and **lsp**. The **lsp** statement specifies the LSP to be used as the actual egress protection LSP.

Figure 68: Egress Protection LSP Configured from Router PE2 to Router PE3



Pseudowires are configured along two paths, one from router PE1 to router PE2 (pseudowire 12) and one from router PE1 to router PE3 (pseudowire 13). In the event of a failure on the link between router PE2 and device CE2, traffic is switched to the egress protection LSP configured between router PE2 and router PE3 (the protector PE router):

- Device CE1—Traffic origin
- Router PE1—Ingress PE router
- Router PE2—Egress PE router
- Router PE3—Protector PE router
- Device CE2—Traffic destintation

This example shows how to configure routers PE1, PE2, and PE3.

Egress Protection LSP Configuration

- Step-by-Step Procedure on page 667

CLI Quick Configuration

To quickly configure an egress protection LSP, copy the following commands into a text file, modify the interface configurations to match your equipment, remove any line breaks, and then paste the commands into the CLI. This group of set commands is for router PE1.

```
set protocols rsvp interface ge-3/1/0.1
set protocols rsvp interface ge-3/1/3.2
set protocols mpls interface ge-3/1/0.1
set protocols mpls interface ge-3/1/3.2
set protocols ospf traffic-engineering
set protocols ospf area 0.0.0.0 interface ge-3/1/0.1
set protocols ospf area 0.0.0.0 interface ge-3/1/3.2
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set protocols ldp interface ge-3/1/0.1
set protocols ldp interface ge-3/1/3.2
set protocols ldp interface lo0.0
set protocols l2circuit neighbor 1.1.1.3 interface fe-3/3/0.1 virtual-circuit-id 32
set protocols l2circuit neighbor 1.1.1.3 interface fe-3/3/0.1 egress-protection
  protector-interface ge-3/1/2.2
set protocols l2circuit neighbor 1.1.1.4 interface ge-3/1/2.2 virtual-circuit-id 33
set policy-options policy-statement load-balance-example then load-balance per-packet
set routing-options router-id 1.1.1.2
set routing-options forwarding-table export load-balance-example
```

To quickly configure an egress protection LSP, copy the following commands into a text file, modify the interface configurations to match your equipment, remove any line breaks, and then paste the commands into the CLI. This group of set commands is for router PE2.

```
[edit]
set protocols rsvp tunnel-services
set protocols rsvp interface ge-2/3/0.3
set protocols rsvp interface ge-2/3/1.4 link-protection
set protocols ldp interface ge-2/3/0.3
set protocols ldp interface ge-2/3/1.4
set protocols ldp interface lo0.0
set protocols ldp upstream-label-assignment
set protocols mpls label-switched-path protected-lsp to 2.2.3.4
set protocols mpls label-switched-path protected-lsp egress-protection
set protocols mpls interface ge-2/3/0.3
set protocols mpls interface ge-2/3/1.4
set protocols ospf traffic-engineering
set protocols ospf area 0.0.0.0 interface ge-2/3/0.3
set protocols ospf area 0.0.0.0 interface ge-2/3/1.4
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set protocols l2circuit neighbor 1.1.1.2 interface ge-2/0/2.5 virtual-circuit-id 23
set protocols l2circuit neighbor 1.1.1.2 interface ge-2/0/2.5 egress-protection protector-pe
  1.1.1.4
set protocols l2circuit neighbor 1.1.1.2 interface ge-2/0/2.5 egress-protection protector-pe
  context-identifier 2.2.3.4
set policy-options policy-statement load-balance-example then load-balance per-packet
set routing-options router-id 1.1.1.3
```

```
set routing-options forwarding-table export load-balance-example
```

To quickly configure an egress protection LSP, copy the following commands into a text file, modify the interface configurations to match your equipment, remove any line breaks, and then paste the commands into the CLI. This group of set commands is for router PE3.

```
set protocols rsvp tunnel-services
set protocols rsvp interface ge-2/0/0.6
set protocols rsvp interface ge-2/0/1.7
set protocols mpls interface ge-2/0/0.6
set protocols mpls interface ge-2/0/1.7
set protocols mpls egress-protection context-identifier 2.2.3.4 protector
set protocols ospf traffic-engineering
set protocols ospf area 0.0.0.0 interface ge-2/0/0.6
set protocols ospf area 0.0.0.0 interface ge-2/0/1.7
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set protocols ldp interface ge-2/0/0.6
set protocols ldp interface ge-2/0/1.7
set protocols ldp interface lo0.0
set protocols ldp upstream-label-assignment
set protocols l2circuit neighbor 1.1.1.2 interface ge-2/0/3.21 virtual-circuit-id 42
set protocols l2circuit neighbor 1.1.1.2 interface ge-2/0/3.21 egress-protection
  protected-l2circuit PW1
set protocols l2circuit neighbor 1.1.1.2 interface ge-2/0/3.21 egress-protection
  protected-l2circuit ingress-pe 1.1.1.2
set protocols l2circuit neighbor 1.1.1.2 interface ge-2/0/3.21 egress-protection
  protected-l2circuit egress-pe 1.1.1.3
set protocols l2circuit neighbor 1.1.1.2 interface ge-2/0/3.21 egress-protection
  protected-l2circuit virtual-circuit-id 31
```

Step-by-Step 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 an egress protection LSP, complete the following steps for router PE1:

1. Configure RSVP. Include the interface linked to router PE2 and the interface linked to router PE3.

   ```
   [edit]
   user@PE1# edit protocols rsvp
   [edit protocols rsvp]
   user@PE1# set interface ge-3/1/0.1
   [edit protocols rsvp]
   user@PE1# set interface ge-3/1/3.2
   ```

2. Configure LDP. Include the interface linked to router PE2, the interface linked to router PE3, and the loopback interface.

   ```
   [edit]
   user@PE1# edit protocols ldp
   [edit protocols ldp]
   user@PE1# set interface ge-3/1/0.1
   [edit protocols ldp]
   ```

```
user@PE1# set interface ge-3/1/3.2
[edit protocols ldp]
user@PE1# set interface lo0.0
```

3. Configure MPLS. Include the interface linked to router PE2 and the interface linked to router PE3.

```
[edit]
user@PE1# edit protocols mpls
[edit protocols mpls]
user@PE1# set interface ge-3/1/0.1
[edit protocols mpls]
user@PE1# set interface ge-3/1/3.2
```

4. Configure OSPF. Include the interface linked to router PE2, the interface linked to router PE3, and the loopback interface in the configuration for the OSPF area.

```
[edit]
user@PE1# edit protocols ospf
[edit protocols ospf]
user@PE1# set interface traffic-engineering
[edit protocols ospf]
user@PE1# set area 0.0.0.0 interface ge-3/1/0.1
[edit protocols ospf]
user@PE1# set area 0.0.0.0 interface ge-3/1/3.2
[edit protocols ospf]
user@PE1# set area 0.0.0.0 interface lo0.0 passive
```

5. Configure Layer 2 circuits to use the egress protection LSP to protect against a link failure to device CE1.

```
set protocols l2circuit neighbor 1.1.1.4 interface ge-3/1/2.2 virtual-circuit-id 33
[edit]
user@PE1# edit protocols l2circuit
[edit protocols l2circuit]
user@PE1# set neighbor 1.1.1.3 interface fe-3/3/0.1 virtual-circuit-id 32
[edit protocols l2circuit]
user@PE1# edit neighbor 1.1.1.3
[edit protocols l2circuit neighbor 1.1.1.3]
user@PE1# interface fe-3/3/0.1 egress-protection protector-interface ge-3/1/2.2
[edit protocols l2circuit]
user@PE1# set neighbor 1.1.1.4 interface ge-3/1/2.2 virtual-circuit-id 33
```

6. Configure a load balancing policy.

```
[edit]
user@PE1# set policy-options policy-statement load-balance-example then
  load-balance per-packet
```

7. Configure the routing options to export routes based on the load balancing policy.

```
[edit]
user@PE1# set routing-options router-id 1.1.1.2
[edit]
user@PE1# set routing-options forwarding-table export load-balance-example
```

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

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 an egress protection LSP, complete the following steps for router PE2:

1. Configure RSVP. Include the interface linked to the ingress PE router and the interface linked to the CE device.

```
[edit]
user@PE2# edit protocols rsvp
[edit protocols rsvp]
user@PE2# set tunnel-services
[edit protocols rsvp]
user@PE2# set interface ge-2/3/0.3
[edit protocols rsvp]
user@PE2# set interface ge-2/3/1.4 link-protection
```

2. Configure LDP. Include the interface linked to the ingress PE router and the interface linked to the CE device.

```
[edit]
user@PE2# edit protocols ldp
[edit protocols ldp]
user@PE2# set interface ge-2/3/0.3
[edit protocols ldp]
user@PE2# set interface ge-2/3/1.4
[edit protocols ldp]
user@PE2# set interface lo0.0
[edit protocols ldp]
user@PE2# set upstream-label-assignment
```

3. Configure MPLS and the LSP which acts as the egress protection LSP.

```
[edit]
user@PE2# edit protocols mpls
[edit protocols mpls]
user@PE2# set interface ge-2/3/0.3
[edit protocols mpls]
user@PE2# set interface ge-2/3/1.4
[edit protocols mpls]
user@PE2# set label-switched-path protected-lsp to 2.2.3.4
[edit protocols mpls]
user@PE2# set label-switched-path protected-lsp egress-protection
```

4. Configure OSPF.

```
[edit]
user@PE2# edit protocols ospf
[edit protocols ospf]
user@PE2# set interface traffic-engineering
[edit protocols ospf]
user@PE2# set interface area 0.0.0.0 interface ge-2/3/0.3
[edit protocols ospf]
user@PE2# set interface area 0.0.0.0 interface ge-2/3/1.4
[edit protocols ospf]
user@PE2# set interface area 0.0.0.0 interface lo0.0 passive
```

5. Configure the Layer 2 circuit to use the egress protection LSP.

```
[edit]
user@PE2# edit protocols l2circuit
[edit protocols l2circuit]
user@PE2# set neighbor 1.1.1.2 interface ge-2/0/2.5 virtual-circuit-id 23
[edit protocols l2circuit]
user@PE2# edit neighbor 1.1.1.2
[edit protocols l2circuit neighbor 1.1.1.2]
user@PE2# set interface ge-2/0/2.5 egress-protectionprotector-pe 1.1.1.4
[edit protocols l2circuit neighbor 1.1.1.2]
user@PE2# set interface ge-2/0/2.5 egress-protection protector-pe context-identifier
  2.2.3.4
```

6. Configure a load balancing policy.

```
[edit]
user@PE1# set policy-options policy-statement load-balance-example then
  load-balance per-packet
```

7. Configure the routing options to export routes based on the load balancing policy.

```
[edit]
user@PE2# set routing-options router-id 1.1.1.3
[edit]
user@PE2# set routing-options forwarding-table export load-balance-example
```

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

Step-by-Step Procedure

To configure an egress protection LSP, complete the following steps for router PE3:

1. Configure RSVP. Include the interface linked to the ingress PE router and the interface linked to the CE device.

```
[edit]
user@PE3# edit protocols rsvp
[edit protocols rsvp]
user@PE3# set tunnel-services
[edit protocols rsvp]
user@PE3# set interface ge-2/0/0.6
[edit protocols rsvp]
user@PE3# set interface ge-2/0/1.7
```

2. Configure LDP. Include the interface linked to the ingress PE router and the interface linked to the CE device.

```
[edit]
user@PE3# edit protocols ldp
[edit protocols ldp]
user@PE3# set interface ge-2/0/0.6
[edit protocols ldp]
user@PE3# set interface ge-2/0/1.7
[edit protocols ldp]
user@PE3# set interface lo0.0
[edit protocols ldp]
user@PE3# set upstream-label-assignment
```

3. Configure MPLS and the LSP which acts as the egress protection LSP.

```
[edit]
user@PE3# edit protocols mpls
[edit protocols mpls]
user@PE3# set interface ge-2/0/0.6
[edit protocols mpls]
user@PE3# set interface ge-2/0/1.7
[edit protocols mpls]
user@PE3# set egress-protection context-identifier 2.2.3.4 protector
```

4. Configure OSPF.

```
[edit]
user@PE3# edit protocols ospf
[edit protocols ospf]
user@PE3# set interface traffic-engineering
[edit protocols ospf]
user@PE3# set area 0.0.0.0 interface ge-2/0/0.6
[edit protocols ospf]
user@PE3# set area 0.0.0.0 interface ge-2/0/1.7
[edit protocols ospf]
user@PE3# set area 0.0.0.0 interface lo0.0 passive
```

5. Configure the Layer 2 circuit to use the egress protection LSP.

```
[edit]
user@PE3# edit protocols l2circuit
[edit protocols l2circuit]
user@PE3# set neighbor 1.1.1.2 interface ge-2/0/3.21 virtual-circuit-id 42
[edit protocols l2circuit]
user@PE3# edit neighbor 1.1.1.2
[edit protocols l2circuit neighbor 1.1.1.2]
user@PE3# set interface ge-2/0/3.21 egress-protection protected-l2circuit ingress-pe
  1.1.1.2
[edit protocols l2circuit neighbor 1.1.1.2]
user@PE3# set interface ge-2/0/3.21 egress-protection protected-l2circuit egress-pe
  1.1.1.3
[edit protocols l2circuit neighbor 1.1.1.2]
user@PE3# set interface ge-2/0/3.21 egress-protection
  protected-l2circuitvirtual-circuit-id 31
```

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

Results From configuration mode, confirm your configuration on router PE1 by entering the **show protocols**, **show policy-options**, and **show routing-options** commands. If the output does not display the intended configuration, repeat the configuration instructions in this example to correct it.

```
show protocols
  rsvp {
    interface ge-3/1/0.1;
    interface ge-3/1/3.2;
  }
  mpls {
    interface ge-3/1/0.1;
    interface ge-3/1/3.2;
  }
  ospf {
```

```
    traffic-engineering;
    area 0.0.0.0 {
      interface ge-3/1/0.1;
      interface ge-3/1/3.2;
      interface lo0.0 {
        passive;
      }
    }
  }
  ldp {
    interface ge-3/1/0.1;
    interface ge-3/1/3.2;
    interface lo0.0;
  }
  l2circuit {
    neighbor 1.1.1.3 {
      interface fe-3/3/0.1 {
        virtual-circuit-id 32;
        egress-protection {
          protector-interface ge-3/1/2.2;
        }
      }
    }
    neighbor 1.1.1.4 {
      interface ge-3/1/2.2 {
        virtual-circuit-id 33;
      }
    }
  }
show policy-options
policy-statement load-balance-example {
  then {
    load-balance per-packet;
  }
}
show routing-options
router-id 1.1.1.2;
  forwarding-table {
    export load-balance-example;
  }
```

From configuration mode, confirm your configuration on router PE2 by entering the **show protocols**, **show policy-options**, and **show routing-options** commands. If the output does not display the intended configuration, repeat the configuration instructions in this example to correct it.

```
show protocols
protocols {
  rsvp {
    tunnel-services;
    interface ge-2/3/0.3;
    interface ge-2/3/1.4 {
      link-protection;
    }
  }
  mpls {
```

```
    label-switched-path protected-lsp {
      to 2.2.3.4;
      egress-protection;
    }
    interface ge-2/3/0.3;
    interface ge-2/3/1.4;
  }
  ospf {
    traffic-engineering;
    area 0.0.0.0 {
      interface ge-2/3/0.3;
      interface ge-2/3/1.4;
      interface lo0.0 {
        passive;
      }
    }
  }
  ldp {
    interface ge-2/3/0.3;
    interface ge-2/3/1.4;
    interface lo0.0;
    upstream-label-assignment;
  }
  l2circuit {
    neighbor 1.1.1.2 {
      interface ge-2/0/2.5 {
        virtual-circuit-id 23;
        egress-protection {
          protector-pe 1.1.1.4 context-identifier 2.2.3.4;
        }
      }
    }
  }
}
```

show policy-options

```
policy-options {
  policy-statement load-balance-example {
    then {
      load-balance per-packet;
    }
  }
}
```

show routing-options

```
routing-options {
  router-id 1.1.1.3;
  forwarding-table {
    export load-balance-example;
  }
}
```

From configuration mode, confirm your configuration on router PE3 by entering the **show protocols** command. If the output does not display the intended configuration, repeat the configuration instructions in this example to correct it.

show protocols

```
rsvp {
    tunnel-services;
    interface ge-2/0/0.6;
    interface ge-2/0/1.7;
}
mpls {
    interface ge-2/0/0.6;
    interface ge-2/0/1.7;
    egress-protection {
        context-identifier 2.2.3.4 {
            protector;
        }
    }
}
ospf {
    traffic-engineering;
    area 0.0.0.0 {
        interface ge-2/0/0.6;
        interface ge-2/0/1.7;
        interface lo0.0 {
            passive;
        }
    }
}
ldp {
    interface ge-2/0/0.6;
    interface ge-2/0/1.7;
    interface lo0.0;
    upstream-label-assignment;
}
l2circuit {
    neighbor 1.1.1.2 {
        interface ge-2/0/3.21 {
            virtual-circuit-id 42;
            egress-protection {
                protected-l2circuit PW1 ingress-pe 1.1.1.2 egress-pe 1.1.1.3 virtual-circuit-id 31;
            }
        }
    }
}
```

CHAPTER 29

Summary of Layer 2 Circuit Configuration Statements

The following sections explain the major protocol configuration statements that apply specifically to Layer 2 circuits. The statements are organized alphabetically. Protocols and the statements at the **[edit protocols]** hierarchy level are explained in the *Junos OS Routing Protocols Configuration Guide*.

bandwidth

Syntax bandwidth (*bandwidth* | ct*number bandwidth*);

Hierarchy Level [edit logical-systems *logical-system-name* protocols l2circuit neighbor *address* interface *interface-name*],
[edit protocols l2circuit neighbor *address* interface *interface-name*]

Release Information Statement introduced before Junos OS Release 7.4.

Description Specify bandwidth allocation for a Layer 2 circuit or for the class types of a Layer 2 circuit.

Options ***bandwidth***—Configure the bandwidth in bits per second for the Layer 2 circuit. You cannot configure the bandwidth for the Layer 2 circuit and for the class types at the same time.

ct*number bandwidth*—Configure the bandwidth in bits per second for a class type on the Layer 2 circuit. You can configure bandwidth for up to four class types (**ct0**, **ct1**, **ct2**, **ct3**) per Layer 2 circuit. If you configure the class types, you must configure them in order, starting with class type **ct0**.

Required Privilege Level routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation

- Configuring Bandwidth Allocation and Call Admission Control in Layer 2 Circuits on page 655

community

Syntax

```
community community-name {
    invert-match;
    members community-members;
}
```

Hierarchy Level

[edit logical-systems *logical-system-name* policy-options],
[edit logical-systems *logical-system-name* protocols l2circuit neighbor *address* interface *interface-name*],
[edit logical-systems *logical-system-name* protocols l2circuit neighbor *address* interface *interface-name* backup-neighbor *address*],
[edit logical-systems *logical-system-name* routing-instances *routing-instance-name* protocols vpls neighbor *address* backup-neighbor *address*],
[edit policy-options],
[edit protocols l2circuit neighbor *address* interface *interface-name*],
[edit protocols l2circuit neighbor *address* interface *interface-name* backup-neighbor *address*],
[edit routing-instances *routing-instance-name* protocols vpls neighbor *address* backup-neighbor *address*]

Release Information

Statement introduced before Junos OS Release 7.4.
Hierarchy levels associated with the **backup-neighbor** statement (pseudowire redundancy) added in Junos OS Release 9.2.

Description

Specify the community for the Layer 2 circuit.

Options

invert-match—Invert the results of the community expression match.

members *community-members*—Specify the members of the community.

Required Privilege Level

routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation

- Configuring the Layer 2 Circuit Community on page 651
- Configuring Pseudowire Redundancy on the PE Router on page 34

control-word

Syntax (control-word | no-control-word);

Hierarchy Level [edit logical-systems *logical-system-name* protocols l2circuit neighbor *address* interface *interface-name*],
[edit protocols l2circuit neighbor *address* interface *interface-name*],

Release Information Statement introduced before Junos OS Release 7.4.

Description Specify the control word. The control word is four bytes long and is inserted between the Layer 2 protocol data unit (PDU) being transported and the virtual circuit (VC) label that is used for demultiplexing.

Options **control-word**—Enable the use of the control word.

Default: A null control word is enabled by default. You can also configure the control word explicitly using the **control-word** statement.

no-control-word—Disable the use of the control word.

Required Privilege Level routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation
- Configuring the Control Word for Frame Relay Interfaces on page 643

description

Syntax description *text*;

Hierarchy Level [edit logical-systems *logical-system-name* protocols l2circuit neighbor *address* interface *interface-name*],
[edit protocols l2circuit neighbor *address* interface *interface-name*]

Release Information Statement introduced before Junos OS Release 7.4.

Description Provide a text description for the Layer 2 circuit. If the text includes one or more spaces, enclose the entire text string in quotation marks (" ").

Required Privilege Level routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation
- Configuring the Description on page 19

egress-protection (Layer 2 circuit)

Syntax

```
egress-protection {
  protected-l2circuit {
    egress-pe address;
    ingress-pe address;
    virtual-circuit-id identifier;
  }
  protector-interface interface-name;
  protector-pe address {
    context-identifier identifier;
    lsp lsp-name;
  }
}
```

Hierarchy Level [edit logical-systems *logical-system-name* protocols l2circuit neighbor *address* interface *interface-name*],
[edit protocols l2circuit neighbor *address* interface *interface-name*]

Release Information Statement introduced in Junos OS release 10.4.

Description Configures an egress protection virtual circuit (EPVC).

Options The other statements are explained separately.

Required Privilege Level routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation

- Example: Configuring an Egress Protection LSP for a Layer 2 Circuit on page 664

egress-protection (MPLS)

Syntax

```
egress-protection {
  context-identifier context-id;
}
```

Hierarchy Level [edit logical-systems *logical-system-name* protocols mpls],
[edit logical-systems *logical-system-name* protocols mpls label-switched-path *lsp-name*],
[edit protocols mpls]
[edit protocols mpls label-switched-path *lsp-name*],

Release Information Statement introduced in Junos OS Release 10.4.

Description Enables an Edge Protection Virtual Circuit (EPVC) for the MPLS protocol.

Options context-identifier *context-id*—(Optional) Specify the context identifier using an IPv4 address.

Required Privilege Level routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

encapsulation-type

Syntax encapsulation-type (atm-aal5 | atm-cell | atm-cell-port-mode | atm-cell-vc-mode | atm-cell-vp-mode | cesop | cisco-hdlc | ethernet | ethernet-vlan | frame-relay | frame-relay-port-mode | interworking | ppp | satop-e1 | satop-e3 | satop-t1 | satop-t3);

Hierarchy Level [edit logical-systems *logical-system-name* protocols l2circuit neighbor *address* interface *interface-name*],
[edit protocols l2circuit neighbor *address* interface *interface-name*]

Release Information Statement introduced in Junos OS Release 9.2.

Description Specify the type of Layer 2 traffic transiting the Layer 2 circuit.

Options

atm-aal5—ATM Adaptation Layer (AAL/5)

atm-cell—ATM cell relay

atm-cell-port-mode—ATM cell relay port promiscuous mode

atm-cell-vc-mode—ATM VC cell relay nonpromiscuous mode

atm-cell-vp-mode—ATM virtual path (VP) cell relay promiscuous mode

cesop—CESOP-based Layer 2 circuit

cisco-hdlc—Cisco Systems–compatible HDLC

ethernet—Ethernet

ethernet-vlan—Ethernet VLAN

frame-relay—Frame Relay

frame-relay-port-mode—Frame Relay port mode

interworking—Layer 2.5 interworking

ppp—PPP

satsop-e1—SATSOP-E1-based Layer 2 circuit

satsop-e3—SATSOP-E3-based Layer 2 circuit

satsop-t1—SATSOP-T1-based Layer 2 circuit

satsop-t3—SATSOP-T3-based Layer 2 circuit

Required Privilege Level routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation

- Configuring the Encapsulation Type for the Layer 2 Circuit Neighbor Interface on page 643

end-interface

Syntax

```
end-interface {
    interface interface-name;
    protect-interface interface-name;
}
```

Hierarchy Level

[edit logical-systems *logical-system-name* protocols l2circuit local-switching interface *interface-name*],
[edit protocols l2circuit local-switching interface *interface-name*]

Release Information

Statement introduced before Junos OS Release 7.4.

Description

Specify the end interface for a local interface switch.

The remaining statements are explained separately.

Required Privilege Level

routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration

Related Documentation

- Configuring Local Interface Switching in Layer 2 Circuits on page 649

ignore-encapsulation-mismatch

Syntax

ignore-encapsulation-mismatch;

Hierarchy Level

[edit logical-systems *logical-system-name* protocols l2circuit local-switching interface *interface-name*],
[edit logical-systems *logical-system-name* protocols l2circuit neighbor *address* interface *interface-name*],
[edit protocols l2circuit local-switching interface *interface-name*],
[edit protocols l2circuit neighbor *address* interface *interface-name*]

Release Information

Statement introduced in Junos OS Release 9.2.
Statement extended to support local switching in Junos OS Release 10.4.

Description

Allow a Layer 2 circuit to be established even though the encapsulation configured on the CE device interface does not match the encapsulation configured on the Layer 2 circuit interface.

Required Privilege Level

routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration

Related Documentation

- Enabling the Layer 2 Circuit When the Encapsulation Does Not Match on page 644

ignore-mtu-mismatch

Syntax ignore-mtu-mismatch;

Hierarchy Level [edit logical-systems *logical-system-name* protocols l2circuit local-switching interface *interface-name*],
[edit logical-systems *logical-system-name* protocols l2circuit neighbor *address* interface *interface-name*],
[edit protocols l2circuit local-switching interface *interface-name*],
[edit protocols l2circuit neighbor *address* interface *interface-name*]

Release Information Statement introduced in Junos OS Release 8.5.
Support for remote PE routers added in Junos OS Release 9.2.

Description Ignore the MTU configuration set for the physical interface associated with the local switching interface or with the remote PE router. This allows a Layer 2 circuit to be brought up between two logical interfaces that are defined on physical interfaces with different MTU values.

Required Privilege Level routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration

Related Documentation
- Enabling the Layer 2 Circuit When the MTU Does Not Match on page 644
- Enabling Local Interface Switching When the MTU Does Not Match on page 650

install-nexthop

Syntax install-nexthop (except | lsp *lsp-name* | lsp-regex *lsp-regular-expression*);

Hierarchy Level [edit logical-systems *logical-system-name* policy-options policy-statement *policy-name* term *term-name* then],
[edit policy-options policy-statement *policy-name* term *term-name* then]

Release Information Statement introduced before Junos OS Release 7.4.

Description Select a specific label-switched path (LSP), or select an LSP from a set of similarly named LSPs as the traffic destination for the configured community. Also can prevent the installation of any matching next hops.

Options

except—Prevent the installation of any matching next hops.

lsp *lsp-name*—Configure a specific LSP.

lsp-regex *lsp-regular-expression*—Configure a range of similarly named LSPs. You can use the following wildcard characters when configuring an LSP regular expression:

- Asterisk (*)—Match any characters.
- Period (.)—Match any single digit.

Required Privilege Level routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation

- Configuring the Policy Statement for the Layer 2 Circuit Community on page 652

interface

Syntax

```
interface interface-name {
    bandwidth (bandwidth | ctnumber bandwidth);
    community community-name;
    (control-word | no-control-word);
    description text;
    encapsulation-type type;
    ignore-encapsulation-mismatch;
    ignore-mtu-mismatch;
    mtu mtu-number;
    protect-interface interface-name;
    psn-tunnel-endpoint address;
    virtual-circuit-id identifier;
}
```

Hierarchy Level

[edit logical-systems *logical-system-name* protocols l2circuit local-switching],
[edit logical-systems *logical-system-name* protocols l2circuit neighbor *address*],
[edit protocols l2circuit local-switching],
[edit protocols l2circuit neighbor *address*]

Release Information

Statement introduced before Junos OS Release 7.4.

Description

Interface over which Layer 2 circuit traffic travels.

Options

interface-name—Name of the interface to configure.

The remaining statements are explained separately.

Required Privilege Level

routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation

- Configuring the Neighbor Interface for the Layer 2 Circuit on page 641

l2circuit

Syntax

```
l2circuit {
  local-switching {
    interface interface-name {
      description text;
      end-interface {
        interface interface-name;
        protect-interface interface-name;
      }
      ignore-mtu-mismatch;
      protect-interface interface-name;
    }
  }
  neighbor address {
    interface interface-name {
      bandwidth (bandwidth | ctnumber bandwidth);
      community community-name;
      (control-word | no-control-word);
      description text;
      encapsulation-type type;
      ignore-encapsulation-mismatch;
      ignore-mtu-mismatch;
      mtu mtu-number;
      protect-interface interface-name;
      pseudowire-status-tlv;
      psn-tunnel-endpoint address;
      virtual-circuit-id identifier;
    }
  }
  traceoptions {
    file filename <files number> <size size> <world-readable | no-world-readable>;
    flag flag <flag-modifier> <disable>;
  }
}
```

Hierarchy Level

[edit logical-systems *logical-system-name* protocols],
[edit protocols]

Release Information

Statement introduced before Junos OS Release 7.4.
Statement introduced in Junos OS Release 11.1 for EX Series switches.

Description

Enables a Layer 2 circuit.

The remaining statements are explained separately.

Required Privilege Level

routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation

- Configuring ATM Trunking on Layer 2 Circuits on page 654
- Configuring Bandwidth Allocation and Call Admission Control in Layer 2 Circuits on page 655

- Configuring Interfaces for Layer 2 Circuits on page 640
- Configuring LDP for Layer 2 Circuits on page 650
- Configuring Policies for Layer 2 Circuits on page 651
- Configuring Static Layer 2 Circuits on page 650
- Introduction to Configuring Layer 2 Circuits on page 639
- Tracing Layer 2 Circuit Operations on page 656

local-switching

Syntax

```
local-switching {
  interface interface-name {
    description text;
    end-interface {
      interface interface-name;
      protect-interface interface-name;
    }
    ignore-mtu-mismatch;
    protect-interface interface-name;
  }
}
```

Hierarchy Level [edit logical-systems *logical-system-name* protocols l2circuit],
[edit protocols l2circuit]

Release Information Statement introduced before Junos OS Release 7.4.

Description Configure a local switching interface. A local switching interface allows you to terminate a virtual circuit on the local router.

The remaining statements are explained separately.

Required Privilege Level routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation

- Configuring Local Interface Switching in Layer 2 Circuits on page 649

mtu

Syntax mtu *mtu-number*;

Hierarchy Level [edit logical-systems *logical-system-name* protocols l2circuit neighbor *address* interface *interface-name*],
[edit protocols l2circuit neighbor *address* interface *interface-name*]

Release Information Statement introduced before Junos OS Release 7.4.

Description Configure the MTU to be advertised for the Layer 2 circuit.

Options ***mtu-number***—MTU number to be advertised for the Layer 2 circuit.

Required Privilege Level routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation
- Configuring the MTU Advertised for a Layer 2 Circuit on page 645

neighbor

Syntax

```
neighbor address {
  interface interface-name {
    bandwidth (bandwidth | ctnumber bandwidth);
    community community-name;
    (control-word | no-control-word);
    description text;
    ignore-encapsulation-mismatch;
    ignore-mtu-mismatch;
    mtu mtu-number;
    protect-interface interface-name;
    pseudowire-status-tlv;
    psn-tunnel-endpoint address;
    static {
      incoming-label label;
      outgoing-label label;
    }
    virtual-circuit-id identifier;
  }
}
```

Hierarchy Level [edit logical-systems *logical-system-name* protocols l2circuit],
[edit protocols l2circuit]

Release Information Statement introduced before Junos OS Release 7.4.
Statement introduced in Junos OS Release 11.1 for EX Series switches.

Description Each Layer 2 circuit is represented by the logical interface connecting the local provider edge (PE) router or switch to the local customer edge (CE) router or switch. All the Layer 2 circuits using a particular remote PE router or switch designated for remote CE routers or switches are listed under the **neighbor** statement (neighbor designates the PE router or switch). Each neighbor is identified by its IP address and is usually the end-point destination for the LSP tunnel (transporting the Layer 2 circuit).

Options ***address***—IP address of a neighboring router or switch.

The remaining statements are explained separately.

Required Privilege Level routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation

- Configuring the Neighbor Interface for the Layer 2 Circuit on page 641

no-control-word

See **control-word**

protect-interface

Syntax protect-interface *interface-name*;

Hierarchy Level
[edit logical-systems *logical-system-name* protocols l2circuit neighbor *address* interface *interface-name*],
[edit logical-systems *logical-system-name* protocols l2circuit local-switching interface *interface-name*],
[edit logical-systems *logical-system-name* protocols l2circuit local-switching interface *interface-name* end-interface],
[edit protocols l2circuit local-switching interface *interface-name*],
[edit protocols l2circuit neighbor *address* interface *interface-name*],
[edit protocols l2circuit local-switching interface *interface-name* end-interface]

Release Information Statement introduced before Junos OS Release 7.4.

Description Provide a backup for the protected interface in case of failure. Network traffic uses the primary interface only, as long as the primary interface functions.

Options ***interface-name***—Name of the protect interface to configure.

Required Privilege Level
routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation
- Configuring the Protect Interface on page 645

protected-l2circuit

Syntax

```
protected-l2circuit {
  egress-pe address;
  ingress-pe address;
  virtual-circuit-id identifier;
}
```

Hierarchy Level

[edit logical-systems *logical-system-name* protocols l2circuit neighbor *address* interface *interface-name* **egress-protection**],
[edit protocols l2circuit neighbor *address* interface *interface-name* **egress-protection**]

Release Information

Statement introduced in Junos OS release 10.4.

Description

Configures the protected Layer 2 circuit as part of an egress protection virtual circuit (EPVC).

Options

egress-pe *address*—Specify the address of the egress PE router for the protected Layer 2 circuit.

ingress-pe *address*—Specify the address of the ingress PE router for the protected Layer 2 circuit.

virtual-circuit-id *identifier*—Specify the virtual circuit identifier for the protected Layer 2 circuit.

Required Privilege Level

routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation

- Example: Configuring an Egress Protection LSP for a Layer 2 Circuit on page 664

protector-interface

Syntax protector-interface *interface-name*;

Hierarchy Level [edit logical-systems *logical-system-name* protocols l2circuit neighbor *address* interface *interface-name* egress-protection],
[edit protocols l2circuit neighbor *address* interface *interface-name* egress-protection]

Release Information Statement introduced in Junos OS release 10.4.

Description Configures the protector interface for an egress protection LSP.

Options *interface-name*——Name of the interface used to protect traffic for an egress protection LSP.

Required Privilege Level routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation
- Example: Configuring an Egress Protection LSP for a Layer 2 Circuit on page 664

protector-pe

Syntax
```
protector-pe address {
  context-identifier identifier;
  lsp lsp-name;
  }
}
```

Hierarchy Level [edit logical-systems *logical-system-name* protocols l2circuit neighbor *address* interface *interface-name* egress-protection],
[edit protocols l2circuit neighbor *address* interface *interface-name* egress-protection]

Release Information Statement introduced in Junos OS release 10.4.

Description Configures the protector PE router for an egress protection LSP. Test.

Options *address*——IPv4 address for the protector PE router.

context-identifier *identifier*——Identifies the context for the egress protection LSP.

lsp *lsp-name*——Specifies the LSP for the egress protection LSP.

Required Privilege Level routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation
- Example: Configuring an Egress Protection LSP for a Layer 2 Circuit on page 664

pseudowire-status-tlv

Syntax pseudowire-status-tlv;

Hierarchy Level [edit logical-systems *logical-system-name* protocols l2circuit neighbor *address* interface *interface-name*],
[edit protocols l2circuit neighbor *address* interface *interface-name*],
[edit routing-instances *routing-instance-name* protocols vpls mesh-group *mesh-group-name* neighbor *address*],
[edit routing-instances *routing-instance-name* protocols vpls neighbor *address*]

Release Information Statement introduced in Junos OS Release 10.0.

Description Enables the pseudowire type length variable (TLV). The pseudowire status TLV is used to communicate the status of a pseudowire back and forth between two PE routers. The pseudowire status TLV is configurable for each pseudowire connection and is disabled by default.

Required Privilege Level routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation
- Configuring the Pseudowire Status TLV on page 646

psn-tunnel-endpoint

Syntax psn-tunnel-endpoint *address*;

Hierarchy Level
[edit logical-systems *logical-system-name* protocols l2circuit neighbor *address* interface *interface-name*],
[edit logical-systems *logical-system-name* protocols l2circuit neighbor *address* interface *interface-name* backup-neighbor *address*],
[edit logical-systems *logical-system-name* routing-instances *routing-instance-name* protocols vpls neighbor *address* backup-neighbor *address*],
[edit protocols l2circuit neighbor *address* interface *interface-name*],
[edit protocols l2circuit neighbor *address* interface *interface-name* backup-neighbor *address*],
[edit routing-instances *routing-instance-name* protocols vpls neighbor *address* backup-neighbor *address*]

Release Information Statement introduced before Junos OS Release 7.4.
Hierarchy levels associated with the **backup-neighbor** statement added in Junos OS Release 9.2.

Description Specify the endpoint of the packet switched network (PSN) tunnel on the remote PE router.

Options ***address***—Address for the tunnel endpoint.

Required Privilege Level routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation
- Configuring Layer 2 Circuits over Both RSVP and LDP LSPs on page 646
- Configuring Pseudowire Redundancy on the PE Router on page 34

static

Syntax

```
static {
    incoming-label label;
    outgoing-label label;
}
```

Hierarchy Level

[edit logical-systems *logical-system-name* l2circuit neighbor *address* interface *interface-name*],
[edit logical-systems *logical-system-name* l2circuit neighbor *address* interface *interface-name* backup-neighbor *neighbor*],
[edit protocols l2circuit neighbor *address* interface *interface-name*],
[edit protocols l2circuit neighbor *address* interface *interface-name* backup-neighbor *neighbor*]

Release Information

Statement introduced in Junos OS Release 9.5.

Description

Configures static Layer 2 circuit pseudowires. Static pseudowires are designed for networks that do not support LDP or do not have LDP enabled. You configure a static pseudowire by configuring static values for the in and out labels needed to enable a pseudowire connection.

Options

incoming-label *label*—Configure the incoming label for the static pseudowire.

outgoing-label *label*—Configure the outgoing label for the static pseudowire.

Required Privilege Level

routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation

- Configuring Static Layer 2 Circuits on page 650

traceoptions

Syntax

```
traceoptions {
    file filename <files number> <size size> <world-readable | no-world-readable>;
    flag flag <flag-modifier> <disable>;
}
```

Hierarchy Level [edit logical-systems *logical-system-name* protocols l2circuit],
[edit protocols l2circuit]

Release Information Statement introduced before Junos OS Release 7.4.

Description Trace traffic flowing through a Layer 2 circuit.

Options

disable—(Optional) Disable the tracing operation. You can use this option to disable a single operation when you have defined a broad group of tracing operations, such as **all**.

file *filename*—Name of the file to receive the output of the tracing operation. Enclose the name in quotation marks (" ").

files *number*—(Optional) Maximum number of trace files. When a trace file named ***trace-file*** reaches its maximum size, it is renamed ***trace-file*.0**, then ***trace-file*.1**, and so on, until the maximum number of trace files is reached. Then the oldest trace file is overwritten.

If you specify a maximum number of files, you also must specify a maximum file size with the **size** option.

Range: 2 through 1000 files

Default: 2 files

flag *flag*—Tracing operation to perform. To specify more than one tracing operation, include multiple **flag** statements.

- **connections**—Layer 2 circuit connections (events and state changes)
- **error**—Error conditions
- **fec**—Layer 2 circuit advertisements received or sent by means of LDP
- **topology**—Layer 2 circuit topology changes caused by reconfiguration or advertisements received from other PE routers

flag-modifier—(Optional) Modifier for the tracing flag. You can specify the **detail** modifier if you want to provide detailed trace information.

no-world-readable—(Optional) Do not allow any user to read the log file.

size *size*—(Optional) Maximum size of each trace file, in kilobytes (KB), megabytes (MB), or gigabytes (GB). When a trace file named ***trace-file*** reaches this size, it is renamed ***trace-file*.0**. When the ***trace-file*** again reaches its maximum size, ***trace-file*.0** is renamed

trace-file*.1** and ***trace-file is renamed ***trace-file*.0**. This renaming scheme continues until the maximum number of trace files is reached. Then the oldest trace file is overwritten.

If you specify a maximum file size, you also must specify a maximum number of trace files with the **files** option.

Syntax: ***x*k** to specify kilobytes, ***x*m** to specify megabytes, or ***x*g** to specify gigabytes

Range: 10 KB through the maximum file size supported on your system

Default: 1 MB

world-readable—(Optional) Allow any user to read the log file.

Required Privilege Level
routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation
- Tracing Layer 2 Circuit Operations on page 656

virtual-circuit-id

Syntax
virtual-circuit-id *identifier*;

Hierarchy Level
[edit logical-systems *logical-system-name* protocols l2circuit neighbor *address* interface *interface-name*],
[edit logical-systems *logical-system-name* protocols l2circuit neighbor *address* interface *interface-name* backup-neighbor *address*],
[edit protocols l2circuit neighbor *address* interface *interface-name*],
[edit protocols l2circuit neighbor *address* interface *interface-name* backup-neighbor *address*]

Release Information
Statement introduced before Junos OS Release 7.4. Hierarchy levels for **backup-neighbor** (pseudowire redundancy) added in Junos OS Release 9.2.
Statement introduced in Junos OS Release 11.1 for EX Series switches.

Description
Uniquely identify a Layer 2 circuit for either a regular pseudowire or a redundant pseudowire.

Options
identifier—1 through 4,294,967,295

Required Privilege Level
routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation
- Configuring the Virtual Circuit ID on page 647
- Configuring Pseudowire Redundancy on the PE Router on page 34

PART 8

Indexes

Index

Symbols

A

B

C

D

E

F

G

M

N

O

P

R

S

T

U

V

W

Y

Index of Statements and Commands

W

