JUNIPER NETWORKS

Junos® OS

VPN Overview and Common Configuration

Release
12.1

Published: 2012-03-13

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

Junos® OS VPN Overview and Common Configuration
12.1

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

YEAR 2000 NOTICE

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

END USER LICENSE AGREEMENT

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

Table of Contents

About the Documentation xi
Documentation and Release Notes xi
Using the Examples in This Manual xi
Merging a Full Example xii
Merging a Snippet xii
Documentation Conventions xiii
Documentation Feedback xiv
Requesting Technical Support xv
Self-Help Online Tools and Resources xv
Opening a Case with JTAC xvi

Part 1 Overview

Chapter 1 **Introduction to VPNs 3**

Types of VPNs 3
Layer 2 VPNs 4
Layer 3 VPNs 4
VPLS 4
Virtual-Router Routing Instances 5
VPNs and Class of Service 6
VPNs and Logical Systems 6
VPN Graceful Restart 7
Redundant Pseudowires for Layer 2 Circuits and VPLS 8
Types of Redundant Pseudowire Configurations 8
Pseudowire Failure Detection 9
BFD Support for VCCV for Layer 2 VPNs, Layer 2 Circuits, and VPLS 10
Chained Composite Next Hops for Transit Devices 10

Chapter 2 **Introduction to Configuring VPNs 13**

Configuring an IGP on the PE and P Routers 13
Rewriting Markers and VPNs 13
Transmitting Nonstandard BPDUs 14
Pinging VPNs, VPLS, and Layer 2 Circuits 14
Setting the Forwarding Class of the Ping Packets 15
Pinging a VPLS Routing Instance 15

Part 2 Configuration

Chapter 3 Configuring VPNs 19

Configuring the Signaling Protocol on PE Routers in VPNs 19
Using LDP for VPN Signaling 20
Using RSVP for VPN Signaling 21
Configuring IBGP Sessions Between PE Routers in VPNs 23
Configuring Routing Instances on PE Routers in VPNs 24
Configuring the Routing Instance Name for a VPN 25
Configuring the Description 25
Configuring the Instance Type 25
Configuring Interfaces for VPN Routing 26
General Configuration for VPN Routing 26
Configuring Interfaces for Layer 3 VPNs 27
Configuring Interfaces for Carrier-of-Carriers VPNs 27
Configuring Unicast RPF on VPN Interfaces 27
Configuring the Route Distinguisher 28
Configuring Automatic Route Distinguishers 28
Configuring Policies for the VRF Table on PE Routers in VPNs 29
Configuring the Route Target 29
Configuring the Route Origin 30
Configuring an Import Policy for the PE Router's VRF Table 31
Configuring an Export Policy for the PE Router's VRF Table 32
Applying Both the VRF Export and the BGP Export Policies 34
Configuring a VRF Target 34
Configuring BGP Route Target Filtering in VPNs 35
BGP Route Target Filtering Overview 35
Configuring BGP Route Target Filtering for VPNs 36
Configuring Virtual-Router Routing Instances in VPNs 37
Configuring a Routing Protocol Between the Service Provider Routers 37
Configuring Logical Interfaces Between Participating Routers 38
Configuring Graceful Restart for VPNs 39
Configuring Redundant Pseudowires for Layer 2 Circuits and VPLS 39
Configuring Pseudowire Redundancy on the PE Router 40
Configuring the Switchover Delay for the Pseudowires 41
Configuring a Revert Time for the Redundant Pseudowire 41
Configuring BFD for VCCV for Layer 2 VPNs, Layer 2 Circuits, and VPLS 41
Configuring Aggregate Labels for VPNs 42
Pinging a Layer 2 VPN 43
Pinging a Layer 3 VPN 44
Pinging a Layer 2 Circuit 44
Configuring Path MTU Checks for VPNs 44
Enabling Path MTU Checks for a VPN Routing Instance 45
Assigning an IP Address to the VPN Routing Instance 45
Enabling Unicast Reverse-Path Forwarding Check for VPNs 45

Chapter 4 VPN Examples 47

Example: BGP Route Target Filtering for VPNs 47
Example: BGP Route Target Filtering for VPNs 49
Configure BGP Route Target Filtering on Router PE1 49
Configure BGP Route Target Filtering on Router PE2 51
Configure BGP Route Target Filtering on the Route Reflector 53
Configure BGP Route Target Filtering on Router PE3 55
Route Origin for VPNs 57
Configuring the Site of Origin Community on CE Router A 58
Configuring the Community on CE Router A 58
Applying the Policy Statement on CE Router A 59
Configuring the Policy on PE Router D 59
Configuring the Community on PE Router D 60
Applying the Policy on PE Router D 60

Part 3 Administration

Chapter 5 VPN References 63

Routers in a VPN 63
VPN Terminology 63

Chapter 6 Summary of VPN Configuration Statements 65

aggregate-label 65
backup-neighbor 66
description 67
family route-target 68
graceful-restart 69
instance-type 70
interface 71
no-forwarding 71
revert-time 72
route-distinguisher 73
route-distinguisher-id 74
switchover-delay 75
unicast-reverse-path 76
vpn-apply-export 76
vrf-export 77
vrf-import 78
vrf-mtu-check 78
vrf-target 79

Part 4 Index

Index 83

List of Figures

Part 1 Overview

Chapter 1 Introduction to VPNs 3

Figure 1: Logical Interface per Router in a Virtual-Router Routing Instance 6

Part 2 Configuration

Chapter 4 VPN Examples 47

Figure 2: BGP Route Target Filtering Enabled for a Group of VPNs 49

Figure 3: Network Topology of Site of Origin Example 57

Part 3 Administration

Chapter 5 VPN References 63

Figure 4: Routers in a VPN 63

List of Tables

About the Documentation xi

Table 1: Notice Icons xiii
Table 2: Text and Syntax Conventions xiii

About the Documentation

- Documentation and Release Notes on page xi
- Using the Examples in This Manual on page xi
- Documentation Conventions on page xiii
- Documentation Feedback on page xiv
- Requesting Technical Support on page xv

Documentation and 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/.

If the information in the latest release notes differs from the information in the documentation, follow the product Release Notes.

Juniper Networks Books publishes books by Juniper Networks engineers and subject matter experts. These books go beyond the technical documentation to explore the nuances of network architecture, deployment, and administration. The current list can be viewed at http://www.juniper.net/books.

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. The example does not become active until you commit the 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 xiii 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 xiii 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*

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

Convention	Description	Examples
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.	
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 post-sales 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/
- Search for known bugs: http://www2.juniper.net/kb/
- 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, see http://www.juniper.net/support/requesting-support.html .

PART 1

Overview

- Introduction to VPNs on page 3
- Introduction to Configuring VPNs on page 13

CHAPTER 1

Introduction to VPNs

- Types of VPNs on page 3
- VPNs and Class of Service on page 6
- VPNs and Logical Systems on page 6
- VPN Graceful Restart on page 7
- Redundant Pseudowires for Layer 2 Circuits and VPLS on page 8
- BFD Support for VCCV for Layer 2 VPNs, Layer 2 Circuits, and VPLS on page 10
- Chained Composite Next Hops for Transit Devices on page 10

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 4
- Layer 3 VPNs on page 4
- VPLS on page 4
- Virtual-Router Routing Instances on page 5

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.

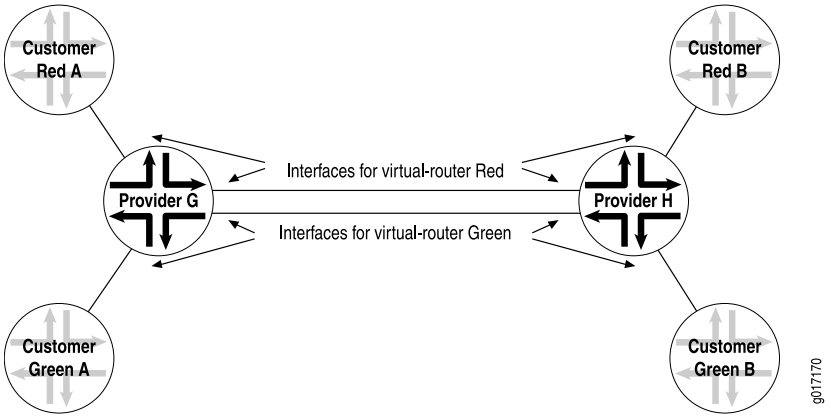
Virtual-Router Routing Instances

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 1 on page 6 shows how this works. The service provider routers G and H are configured for virtual-router routing instances Red and Green. Each service provider router is directly connected to two local customer routers, one in each routing instance. The service provider routers are also connected to each other over the service provider network. These routers need four logical interfaces: a logical interface to each of the locally connected customer routers and a logical interface to carry traffic between the two service provider routers for each virtual-router instance.

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



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

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

VPNs and 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 39.

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

- Types of Redundant Pseudowire Configurations on page 8
- Pseudowire Failure Detection on page 9

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.

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

Bidirectional Forwarding Detection (BFD) support for virtual circuit connectvity verification (VCCV) on MX Series devices enables you to configure a control channel for a pseudowire, in addition to the corresponding operations, administration, 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 RFC 5885, *Bidirectional Forwarding Detection (BFD) for the Pseudowire Virtual Circuit Connectivity Verification (VCCV)*. Alternatively, you can 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 a 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).

Starting with Release 12.1, Junos OS introduces a distributed model for the BFD for VCCV. Unlike in previous releases where the BFD for VCCV followed a Routing Engine-based implementation, in Release 12.1 and later, the BFD for VCCV follows a distributed implementation over PIC concentrators, such as DPC, FPC, and MPC.

In Junos OS Release 12.1 and later, the periodic packet management process (ppmd) on the PIC concentrators handles the periodic packet management (send and receive) for BFD for VCCV. This enables Junos OS to create more BFD for VCCV sessions, and to reduce the time taken for error detection. Similarly, the distributed implementation improves the performance of Routing Engines because the Routing Engine resources used for BFD for VCCV implementation become available for Routing Engine-related applications when the BFD for VCCV-related processing moves to the PIC concentrators. The distributed BFD for VCCV implementation also enables the BFD for VCCV sessions to remain across graceful restarts.

Related Documentation

- Configuring BFD for VCCV for Layer 2 VPNs, Layer 2 Circuits, and VPLS on page 41

Chained Composite Next Hops for Transit Devices

The Juniper Networks PTX Series Packet Transport Switch is principally designed to handle large volumes of transit traffic in the core of large networks. Chained composite next hops help to facilitate this capability by allowing a packet transport switch to process much larger volumes of routes. A chained composite next hop allows the packet transport switch to direct sets of routes sharing the same destination to a common forwarding next hop, rather than having each route also include the destination. In the event that a network destination is changed, rather than having to update all of the routes sharing that destination with the new information, just the shared forwarding next hop is updated with the new information. The chained composite next hops continue to point to this forwarding next hop which now contains the new destination.

When the next hops for MPLS LSPs are created on packet transport switches, the tag information corresponding to the inner-most MPLS label is extracted into a chained composite next hop. The chained composite next hop is stored in the ingress PFE. The chained composite next hop points to a next hop called the forwarding next hop that resides on the egress PFE. The forwarding next hop contains all of the other information (all of the labels except for the inner-most labels; and the IFA/IP information corresponding to the actual next hop node). Many chained composite next hops can share the same forwarding next hop. Additionally, separating the label from the forwarding next hop and storing it on the ingress PFE (within the chained composite next hop) helps to conserve egress PFE memory by reducing the number of rewrite strings stored on the egress PFE.

On PTX Series Packet Transport Switches, chained composite next hops are enabled by default for the following MPLS and VPN protocols and applications:

- Labeled BGP
- Layer 2 VPNs
- Layer 3 VPNs
- LDP
- MPLS
- Point-to-Multipoint LSPs
- RSVP
- Static LSPs

Related Documentation

- Accepting BGP Updates with Unique Inner VPN Labels in Layer 3 VPNs

CHAPTER 2

Introduction to Configuring VPNs

- Configuring an IGP on the PE and P Routers on page 13
- Rewriting Markers and VPNs on page 13
- Transmitting Nonstandard BPDUs on page 14
- Pinging VPNs, VPLS, and Layer 2 Circuits on page 14
- Setting the Forwarding Class of the Ping Packets on page 15
- Pinging a VPLS Routing Instance on page 15

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*.

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 15).

You issue the **ping mpls** command from the ingress PE router of the VPN or Layer 2 circuit to the egress PE router of the same VPN or Layer 2 circuit. When you execute the **ping mpls** command, echo requests are sent as MPLS packets.

The payload is a User Datagram Protocol (UDP) packet forwarded to the address **127.0.0.1**. The contents of this packet are defined in RFC 4379, *Detecting Multi-Protocol Label Switched (MPLS) Data Plane Failures*. The label and interface information for building and sending this information as an MPLS packet is the same as for standard VPN traffic, but the time-to-live (TTL) of the innermost label is set to 1.

When the echo request arrives at the egress PE router, the contents of the packet are checked, and then a reply that contains the correct return is sent by means of UDP. The PE router sending the echo request waits to receive an echo reply after a timeout of 2 seconds (you cannot configure this value).

You must configure MPLS at the **[edit protocols mpls]** hierarchy level on the egress PE router (the router receiving the MPLS echo packets) to be able to ping the VPN or Layer 2 circuit. You must also configure the address **127.0.0.1/32** on the egress PE router's **lo0** interface. If this is not configured, the egress PE router does not have this forwarding entry and therefore simply drops the incoming MPLS pings.

The **ping mpls** command has the following limitations:

- You cannot ping an IPv6 destination prefix.
- You cannot ping a VPN or Layer 2 circuit from a router that is attempting a graceful restart.
- You cannot ping a VPN or Layer 2 circuit from a logical system.

You can also determine whether an LSP linking two PE routers in a VPN is up by pinging the end point address of the LSP. The command you use to ping an MPLS LSP end point is **ping mpls lsp-end-point** *address*. This command tells you what type of LSP (RSVP or LDP) terminates at the address specified and whether that LSP is up or down.

For a detailed description of this command, see the *Junos Routing Protocols and Policies Command Reference*.

Setting the Forwarding Class of the Ping Packets

When you execute the **ping mpls** command, the ping packets forwarded to the destination include MPLS labels. It is possible to set the value of the forwarding class for these ping packets by using the **exp** option with the **ping mpls** command. For example, to set the forwarding class to 5 when pinging a Layer 3 VPN, issue the following command:

```
ping mpls l3vpn westcoast source 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.

PART 2

Configuration

- Configuring VPNs on page 19
- VPN Examples on page 47

CHAPTER 3

Configuring VPNs

- Configuring the Signaling Protocol on PE Routers in VPNs on page 19
- Configuring IBGP Sessions Between PE Routers in VPNs on page 23
- Configuring Routing Instances on PE Routers in VPNs on page 24
- Configuring Policies for the VRF Table on PE Routers in VPNs on page 29
- Configuring BGP Route Target Filtering in VPNs on page 35
- Configuring Virtual-Router Routing Instances in VPNs on page 37
- Configuring Graceful Restart for VPNs on page 39
- Configuring Redundant Pseudowires for Layer 2 Circuits and VPLS on page 39
- Configuring BFD for VCCV for Layer 2 VPNs, Layer 2 Circuits, and VPLS on page 41
- Configuring Aggregate Labels for VPNs on page 42
- Pinging a Layer 2 VPN on page 43
- Pinging a Layer 3 VPN on page 44
- Pinging a Layer 2 Circuit on page 44
- Configuring Path MTU Checks for VPNs on page 44
- Enabling Unicast Reverse-Path Forwarding Check for VPNs on page 45

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 20
- Using RSVP for VPN Signaling on page 21

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 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 25
- Configuring the Description on page 25
- Configuring the Instance Type on page 25
- Configuring Interfaces for VPN Routing on page 26
- Configuring the Route Distinguisher on page 28
- Configuring Automatic Route Distinguishers on page 28

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 26
- Configuring Interfaces for Layer 3 VPNs on page 27
- Configuring Interfaces for Carrier-of-Carriers VPNs on page 27
- Configuring Unicast RPF on VPN Interfaces on page 27

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 29
- Configuring BGP Route Target Filtering in VPNs on page 35

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 29
- Configuring the Route Origin on page 30
- Configuring an Import Policy for the PE Router's VRF Table on page 31
- Configuring an Export Policy for the PE Router's VRF Table on page 32
- Applying Both the VRF Export and the BGP Export Policies on page 34
- Configuring a VRF Target on page 34

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 31.

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 32.

See "Route Origin for VPNs" on page 57 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. If you change the **community add** statement to the **community set** statement, the router at the egress of the Layer 2 VPN link might drop the connection.

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

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

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

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

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

For more information about creating policies, see the *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 32, 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 on page 35
- Configuring BGP Route Target Filtering for VPNs on page 36

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

- Example: BGP Route Target Filtering for VPNs on page 49
- Example: BGP Route Target Filtering for VPNs on page 47
- Route Origin for VPNs on page 57

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 on page 37
- Configuring Logical Interfaces Between Participating Routers on page 38

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 8.

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

- Configuring Pseudowire Redundancy on the PE Router on page 40
- Configuring the Switchover Delay for the Pseudowires on page 41
- Configuring a Revert Time for the Redundant Pseudowire on page 41

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.

Related Documentation

- BFD Support for VCCV for Layer 2 VPNs, Layer 2 Circuits, and VPLS on page 10

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*.

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.

Related Documentation

- Example: Configuring MPLS-Based Layer 2 VPNs

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.

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 45
- Assigning an IP Address to the VPN Routing Instance on page 45

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 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 4

VPN Examples

- Example: BGP Route Target Filtering for VPNs on page 47
- Example: BGP Route Target Filtering for VPNs on page 49
- Route Origin for VPNs on page 57

Example: BGP Route Target Filtering for VPNs

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

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

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

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

```
[edit]
routing-instances {
    vpn1 {
        instance-type vrf;
```

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

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

```
user@host> show route table bgp.rtarget.0
 bgp.rtarget.0: 4 destinations, 6 routes (4 active, 0 holddown, 0 hidden)
 + = Active Route, - = Last Active, * = Both
200:200:101/96
                    *[RTarget/5] 00:10:00
                       Local
200:200:102/96
                    *[RTarget/5] 00:10:00
                       Local
200:200:103/96
                    *[BGP/170] 00:09:48, localpref 100, from 10.255.14.174
                       AS path: I
                     > t3-0/0/0.0
200:200:104/96
                    *[BGP/170] 00:09:48, localpref 100, from 10.255.14.174
                       AS path: I
                     > t3-0/0/0.0
```

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

AS number:route target extended community/length

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

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

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

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

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

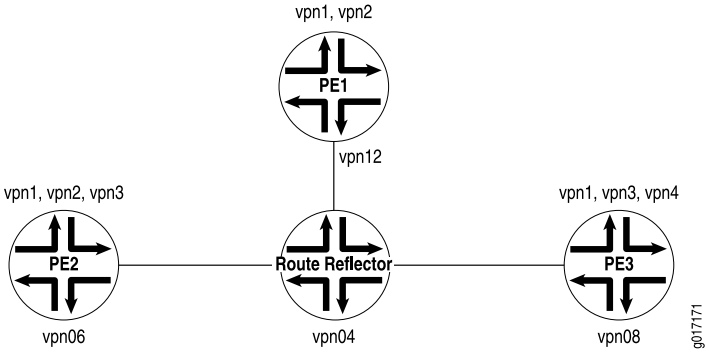
- **200:200:103/96**—Tells the local router that routes tagged with this community (**200:103**) should be advertised to peer **10.255.14.174** through **t3-0/0/0.0**
- **200:200:104/96**—Tells the local router that routes tagged with this community (**200:104**) should be advertised to peer **10.255.14.174**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

Example: 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 2 on page 49 illustrates the topology for a network configured with BGP route target filtering for a group of VPNs.

Figure 2: 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 49
- Configure BGP Route Target Filtering on Router PE2 on page 51
- Configure BGP Route Target Filtering on the Route Reflector on page 53
- Configure BGP Route Target Filtering on Router PE3 on page 55

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:

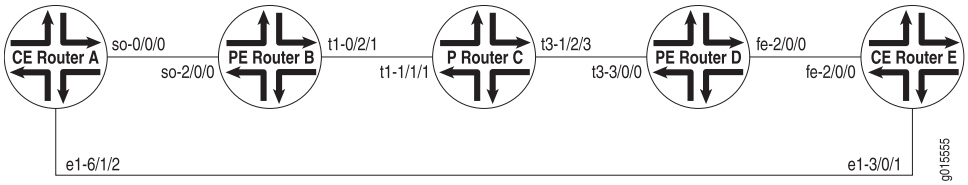
```
user@host> show route table bgp.rtarget.0
bgp.rtarget.0: 4 destinations, 7 routes (4 active, 0 holddown, 0 hidden)
 + = Active Route, - = Last Active, * = Both

 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 3 on page 57.

Figure 3: 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 58
- Configuring the Community on CE Router A on page 58
- Applying the Policy Statement on CE Router A on page 59
- Configuring the Policy on PE Router D on page 59
- Configuring the Community on PE Router D on page 60
- Applying the Policy on PE Router D on page 60

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;
      }
    }
  }
}
```

PART 3

Administration

- VPN References on page 63
- Summary of VPN Configuration Statements on page 65

CHAPTER 5

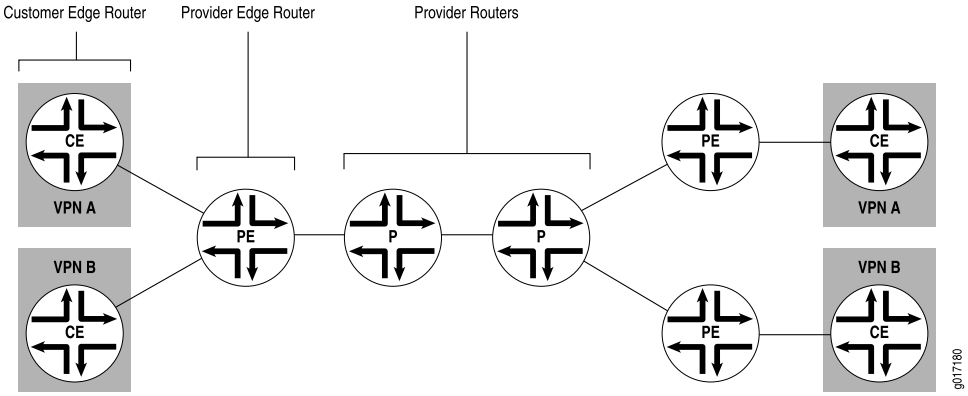
VPN References

- Routers in a VPN on page 63
- VPN Terminology on page 63

Routers in a VPN

Figure 4 on page 63 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 4: 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.

CHAPTER 6

Summary of VPN Configuration Statements

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 42

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
- psn-tunnel-endpoint
- virtual-circuit-id
- Configuring Pseudowire Redundancy on the PE Router on page 40

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 25

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 36

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 39

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 25
- 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 70
- Configuring Interfaces for VPN Routing on page 26

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 37

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 39

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 logical-systems *logical-system-name* routing-instances *routing-instance-name* protocols vpls mesh-group *mesh-group-name*],
[edit routing-instances *routing-instance-name*],
[edit routing-instances *routing-instance-name* protocols vpls mesh-group *mesh-group-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 28
- 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 28
- Example: BGP Route Target Filtering for VPNs on page 49

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 41

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 45

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 34

vrf-export

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

Hierarchy Level
[edit logical-systems *logical-system-name* routing-instances *routing-instance-name*],
[edit logical-systems *logical-system-name* routing-instances *routing-instance-name* protocols vpls mesh-group *mesh-group-name*]
[edit routing-instances *routing-instance-name*]
[edit routing-instances *routing-instance-name* protocols vpls mesh-group *mesh-group-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 70
- Configuring an Export Policy for the PE Router's VRF Table on page 32

vrf-import

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

Hierarchy Level [edit logical-systems *logical-system-name* routing-instances *routing-instance-name*],
[edit logical-systems *logical-system-name* routing-instances *routing-instance-name* protocols vpls mesh-group *mesh-group-name*]
[edit routing-instances *routing-instance-name*]
[edit routing-instances *routing-instance-name* protocols vpls mesh-group *mesh-group-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 70
- Configuring an Import Policy for the PE Router's VRF Table on page 31

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 44

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 logical-systems *logical-system-name* routing-instances *routing-instance-name* protocols vpls mesh-group *mesh-group-name*]
[edit routing-instances *routing-instance-name*],
[edit routing-instances *routing-instance-name* protocols vpls mesh-group *mesh-group-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 34

PART 4

Index

- Index on page 83

Index

Symbols

#, comments in configuration statements....xiv
( ), in syntax descriptions....xiv
< >, in syntax descriptions....xiv
[ ], in configuration statements....xiv
{ }, in configuration statements....xiv
| (pipe), in syntax descriptions....xiv

A

aggregate-label statement....65
    usage guidelines....42
automatic route distinguisher....28
autonomous system number
    route distinguisher....28

B

backup-neighbor statement....66
    usage guidelines....40
BFD
    VCCV....10
BFD for VCCV....10
    Layer 2 VPNs Layer 2 circuits, VPLS....41
BGP
    route target filtering....36, 49
BPDUs, nonstandard....14
braces, in configuration statements....xiv
brackets
    angle, in syntax descriptions....xiv
    square, in configuration statements....xiv

C

CE devices, routers or switches....63
chained composite next hops....10
class of service *See* CoS
comments, in configuration statements....xiv
communities
    regular expressions....32
control-channel statement
    usage guidelines....41
conventions
    text and syntax....xiii
CoS
    VPNs....6
curly braces, in configuration statements....xiv
customer edge devices or routers *See* CE devices
customer support....xv
    contacting JTAC....xv

D

description statement....67
documentation
    comments on....xiv
DoS attack....45

E

export policy, VRF....32

F

family inet-vpn....23
family inet6-vpn....23
family l2vpn....23
family route-target statement....68
    usage guidelines....36
font conventions....xiii

G

graceful-restart statement....69

I

import communities
    regular expressions....32
import policy, VRF....31
instance-type statement....70
    usage guidelines....25
interface statement
    VPNs....71
        usage guidelines....26
IP spoofing....45

L

Layer 2 circuits
    BFD for VCCV....41
    redundant pseudowires....39
Layer 2 VPNs
    BFD for VCCV....41
    overview....4
Layer 3 VPNs
    overview....4

Layer 2 circuits
BPDUs....14
ping command....14
Layer 2 VPNs
BPDUs....14
ping command....14
Layer 3 VPNs
ping command....14
logical systems
VPNs....6
logical-router *See* logical-system
logical-routers *See* logical-systems

M
manuals
comments on....xiv
MPLS
chained composite next hops....10

N
no-forwarding statement....71
usage guidelines....37
nonstandard BPDUs....14

P
P routers....63
parentheses, in syntax descriptions....xiv
path MTU check, VPNs....44
PE routers....64
ping command
usage guidelines
Layer 2 circuits....14
VPLS....14
VPNs....14
ping-interval statement
usage guidelines....41
provider edge routers *See* PE routers
provider routers *See* P routers
pseudowire redundancy
failure detection....9
PTX Series Packet Transport Switch....10

R
redundant pseudowires
configuration....39
overview....8
revert time....41
regular expressions, import communities....32
revert time
redundant pseudowires....41
revert-time statement....72
usage guidelines....41
route distinguisher....28
automatic....28
autonomous system number....28
route reflectors
BGP route target filtering....35, 49
route target filtering, BGP....36
route-distinguisher statement....73
usage guidelines....28
route-distinguisher-id statement....74
usage guidelines....28
route-target statement
usage guidelines....49
routing engine, sampling....26
routing instance name....25
routing instance type....25

S
support, technical *See* technical support
switchover-delay statement....75
usage guidelines....41
syntax conventions....xiii

T
technical support
contacting JTAC....xv

U
unicast RPF
VPNs....45
unicast-reverse-path statement....76
usage guidelines....45

V
VCCV
BFD....10
virtual-router routing instance
overview....5
VPLS
BFD for VCCV....41
overview....4
ping command....14
redundant pseudowires....39
vpn-apply-export statement....76

VPNs
CE devices....63
chained composite next hops....10
CoS....6
export policy....32
import policy....31
interfaces....26
logical systems....6
P routers....63
path MTU check....44
PE routers....64
route distinguisher....28
automatic....28
routing instance name....25
routing instances
path MTU check....45
unicast RPF....27, 45
VRF
export policy....32
import policy....31
regular expressions....32
vrf-export statement....77
vrf-import statement....78
vrf-mtu-check statement....78
usage guidelines....44
vrf-target statement....79